LA PRIMA RIVISTA EUROPEA DI SOFTWARE PER PERSONAL COMPUTER

PERSONAL SOFTWARE

ANNO 2 N. 4
GENNAIO-FEBBRAIO 1983 L. 3.500

UNA PUBBLICAZIONE DEL GRUPPO EDITORIALE JACKSON

Spedizione in abb. postale Gruppo III/70

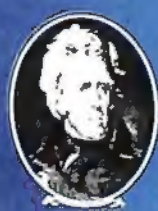
- INTERI IN PRECISIONE MULTIPLA
- GRAFICA TRIDIMENSIONALE CON L'APPLE
- ROUTINE ARITMETICHE PER IL 6502
- I SEGRETI DEI PERSONAL: PET/CBM, ZX80/ZX81, VIC20
- CHE RISCHI CORRE IL VOSTRO SOFTWARE

ANNO 2 N. 4 GENNAIO-FEBBRAIO 1983

PERSONAL SOFTWARE

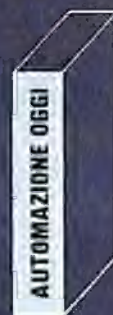
ARTICOLI

RUBRICHE

In questa guida sono riportati i personal computer e i microprocessori di cui si parla negli articoli e nelle rubriche.

Indirizzate tutta la corrispondenza editoriale a

Personal Software, Via Rosellini 12, 20124 Milano

I manoscritti non rischiesti non saranno restituiti. Le opinioni espresse degli autori non sono necessariamente quelle di *Personal Software.*

È disponibile a richiesta una "Guida per gli autori" contenente le informazioni necessarie alla stesura di un articolo o di un programma, oltre a tutte le indicazioni di carattere amministrativo. Richiedetela all'indirizzo indicato.

	l'ELETTRONICA	ELETTRONICA OGGI	AUTOMAZIONE OGGI	ELEKTOR	INFORMATICA OGGI
Numeri	22 numeri	11 numeri	8 numeri	12 numeri	11 numeri
Prezzo	L. 35.000	L. 31.000	L. 19.000	L. 24.500	L. 26.500
anziché	L. 44.000	L. 38.500	L. 24.000	L. 30.000	L. 33.000

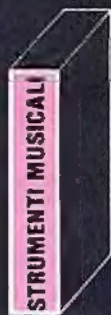
Alcuni esempi

Combinazione	Prezzo
EO + l'E	L. 64.000
EO + AO	L. 48.000
EO + IO	L. 55.500
IO + BT	L. 50.500
IO + l'E	L. 59.500
CW + IO	L. 84.500
BT + PS	L. 52.000
CW + l'E	L. 93.000
EO + l'E + EK	L. 86.500
EO + l'E + IO	L. 88.500
EO + l'E + BT	L. 88.000
IO + BT + PS	L. 76.500
BT + IO + l'E	L. 83.500
EO + l'E + EK + AO	L. 102.500
tutte le riviste ...	L. 266.000

LEGENDA
- l'E = l'ELETTRONICA
- EO = ELETTRONICA OGGI
- AO = AUTOMAZIONE OGGI
- EK = ELEKTOR
- IO = INFORMATICA OGGI
- CW = COMPUTERWORLD
- BT = BIT
- PS = PERSONAL SOFTWARE
- SM = STRUMENTI MUSICALI
- VG = VIDEOGIOCHI

CAMP
ABBON

Per abbonamenti all'estero aumentare
del 50% il prezzo (o i prezzi) scontati di ogni rivista.

38 numeri	11 numeri	10 numeri	10 numeri	10 numeri
L. 60.000	L. 26.000	L. 28.000	L. 24.000	L. 22.000
anzichè	anzichè	anzichè	anzichè	anzichè
~~L. 76.000~~	~~L. 33.000~~	~~L. 35.000~~	~~L. 30.000~~	~~L. 25.000~~
COMPUTERWORLD	BIT	PERSONAL SOFTWARE	STRUMENTI MUSICALI	VIDEO GIOCHI

ABBONAMENTO CUMULATIVO A DUE O PIU' RIVISTE CON SCONTO PARTICOLARE

Tutti coloro che sottoscrivono abbonamenti a due o più riviste godono di un prezzo ulteriormente agevolato, come appare nella seguente tabellina.

Abbonamento a due riviste somma dei prezzi scontati delle due riviste - L. 2.000.

Abbonamento a tre riviste somma dei prezzi scontati delle tre riviste - L. 4.000.

Abbonamento a quattro riviste somma dei prezzi scontati delle quattro riviste - L. 7.000.

Abbonamento a cinque riviste somma dei prezzi scontati delle cinque riviste - L. 10.000.

Abbonamento a sei riviste somma dei prezzi scontati delle sei riviste - L. 13.000.

Abbonamento a sette riviste somma dei prezzi scontati delle sette riviste - L. 16.000.

Abbonamento a otto riviste somma dei prezzi scontati delle otto riviste - L. 20.000.

Abbonamento a nove riviste somma dei prezzi scontati delle nove riviste - L. 25.000.

Abbonamento a dieci riviste somma dei prezzi scontati delle dieci riviste - L. 30.000.

N.B. - **Per sottoscrivere abbonamenti utilizzate il modulo di c.c.p. inserito in questo fascicolo oppure inviate un assegno o un vaglia postale al nostro ufficio abbonamenti.**

AGNA
AMENTI

IL TASTO DEL RISPARMIO.

Software e idee

Mauro Boscarol

Personal Software, appena nata, è già adolescente. Ed ha le sue crisi di crescita. La stiamo curando (anche se non si tratta di malattia, ma di fenomeni fisiologici) ed anzi è quasi guarita. Alcune delle imperfezioni più vistose sono già sparite, altre cose sono migliorate. E d'ora in avanti, speriamo tutti, sarà più puntuale agli appuntamenti.

Il suo termometro sono le lettere dei lettori. Non possiamo rispondere a tutti, ma teniamo conto dei suggerimenti di tutti. C'è chi perora la causa del Texas, chi del computer di Nuova Elettronica, e chi del VIC 20, dello ZX81, dell'Osborne, dell'Olivetti. Naturalmente ci sono delle difficoltà a fare tutto per tutti. Alcuni computer non li abbiamo in redazione: chiediamo ai lettori di aiutarci inviando programmi immediatamente pubblicabili, senza nostra verifica. Altri possiamo invece caricarli, eseguirli, correggerli, ristamparli. Per altri ancora, che pubblichiamo in una certa versione, aspettiamo le conversioni dei lettori per altri computer. Cerchiamo di fare il possibile.

Ma c'è un punto fermo che è il nostro principio fondamentale. Bene i programmi, ma meglio lo scrivere come si fanno, come si risolve un problema, come si superano le difficoltà tecniche. Tra un programma che calcola media e varianza e un buon articolo che illustra le tecniche di ricerca in una tabella, preferiamo il secondo. Tra una caccia al dragone e una illustrazione sulle tecniche della grafica tridimensionale, preferiamo la seconda. Tra un gioco (non intelligente) in cui il computer serve solo da "telecomando" e un gioco (intelligente) in cui il computer può sviluppare una strategia, preferiamo quest'ultimo. Anche se il primo è ad alta risoluzione e a colori, e il secondo senza grafica.

Insomma, il computer ci deve dare una mano per capire, per organizzare il pensiero e i ragionamenti, per farci crescere intellettualmente. Naturalmente, nell'area che ognuno predilige: la grafica, l'organizzazione degli archivi, l'intelligenza artificiale, la musica o la matematica.

Noi, insomma, siamo per le idee. Supportate da una tecnica, naturalmente, ma che è appunto un supporto.

Anche in questo senso Personal Software continua a crescere, e i pareri dei lettori sull'argomento verranno discussi e pubblicati. Come, naturalmente, i loro articoli, che sollecitiamo. Articoli con programmi (meglio) o senza (va bene lo stesso), ma articoli con idee.

In fin dei conti, con un giochino ci si può divertire un'ora, ma un'idea ci può avvincere per sempre.

Ed ora vi presento il "menù" di questo mese.

Antonio Filz, di Trento, ha scritto una bella serie di routine in Basic, adatte a qualunque personal computer, che permettono di eseguire operazioni con numeri interi molto più lunghi di quelli che generalmente sono implementati negli interpreti Basic: con le sue routine si possono eseguire operazioni su numeri con 1000 o 2000 cifre.

Matteo Cerofolini, di Modena, ci propone alcune routine di aritmetica intera sul 6502, ed alcuni test per la verifica della loro velocità d'esecuzione, arrivando a concludere che, se vogliamo avere programmi veloci e che occupino poco spazio, dobbiamo scriverli in linguaggio macchina. Nessuno pensa di inviarci qualche articolo sul linguaggio macchina per principianti?

C'è poi un simpatico programma di Carlo Sintini di Latina: il calcio, scritto per il PET/CBM. Aspettiamo qualche conversione, almeno per il VIC.

Segue un articolo sulla pirateria nel software: le tecniche di protezione più utilizzate e le problematiche connesse.

E infine le rubriche. Nei *Segreti dei personal* Ettore Massimo Albani, di Foggia (ma trascorre la maggior parte del proprio tempo a Padova) spiega come fare in modo che il PET/CBM si autoprogrammi; Enrico Ferreguti di Venezia scrive su come velocizzare un programma per ZX80/ZX81; e infine per i possessori di VIC vi è una simpatica utility.

Nella *Raccolta di routine Basic* questo mese ho scritto io: presento un programma per il controllo del codice fiscale.

E poi, e poi... ma a questo punto vi conviene sfogliare la rivista. ■

ATTENZIONE

Il prossimo numero di
Personal Software
sarà in edicola
il 15 Aprile.

In questa rubrica rispondiamo alle lettere di carattere generale.
Scrivete a
Personal Software
Via Rosellini 12
20124 Milano

Personal Software,
ti seguo da due numeri e francamente mi piaci. Mi spiego: mi piaci come idea, ma per il mio computer non c'è niente... Ho uno Sharp PC-1500 più stampante plotter CE 150, l'ho comprato perché è portatile (ma non dispero di collegarlo a un video) e lo uso prevalentemente per interesse e hobby. In effetti mi piacerebbe ci fosse un angolo per la PC-1500 su P.S., e così vi chiedo se vi posso inviare i miei listati. Si tratta di programmi matematici e hobbistici, e, come tutti i programmatori dilettanti, credo fermamente che siano i migliori. Dunque, ecco le domande:
1) Posso inviare i programmi?
2) Solo listato o anche cassetta? (Tenete conto che le cassette costano un occhio.)
3) C'è un compenso? (Giusta gratificazione per il lavoro compiuto.)

Grazie in anticipo per la risposta, sul giornale o per lettera.

Ernesto de Bernardis
(Catania)

Può senz'altro inviare programmi. Meglio listato e cassetta, così possiamo verificare ed eventualmente sistemare il listato. Il programma verrà esaminato dalla redazione e se sarà giudicato pubblicabile verrà compensato alle tariffe consuete.

Attenzione: un'altra possibilità da tenere in considerazione è la conversione per il suo computer di programmi già apparsi sulla nostra rivista in altre versioni.

Gentile Direzione
Può la Vs. Rivista promuovere gli scambi di programmi (scientifici, applicativi, aziendali, etc.) tra i lettori?
È, secondo me, una iniziativa utile a molti: una specie di mercato interno di personal software.
Con la speranza che la mia proposta sia presa in considerazione, porgo i migliori saluti.

dr. Roberto Martino
Istituto di Fisiologia umana
Università degli Studi di Padova

Già negli scorsi numeri sono apparsi alcuni piccoli annunci. Ma da questo numero la rubrica è più organica.

Nella pubblicazione la precedenza va naturalmente agli annunci relativi a software, libri e riviste. Gli annunci relativi all'hardware verranno anche pubblicati, ma compatibilmente con lo spazio lasciato a disposizione dagli annunci di software.

Per facilitare la lettura abbiamo diviso gli annunci secondo il sistema cui si riferiscono. I lettori che hanno altri suggerimenti sulla rubrica, ce lo facciano sapere.

Spett. Redazione,
credo innanzi tutto sia mio dovere farvi i miei complimenti riguardante la vostra rivista, dato che è forse l'unica in questo settore che riesca nello stesso tempo a fondere la pratica riguardante il notevole numero di programmi; e la teoria nei riguardi degli eccellenti servizi disponibili su di essa.

Non sono un assiduo lettore di nessuna rivista in particolare, ma ne compro varie a seconda del loro contenuto o dell'interesse che suscitano in me. Questa però, devo proprio dirlo, è forse l'unica che ha risvegliato in me l'interesse che nasce da una buona lettura. Devo però fare un piccolo rimprovero (se mi è consentito), ma non rivolto alla vostra rivista in specifico modo, ma rivolto a tutte indistintamente. Su ogni rivista da me fino ad ora sfogliata ho sempre visto listati di programmi per personal e mini ben specifici che non superano i 3 o 4 modelli.

Modelli che vanno dal più piccolo come il Sinclair, al più famoso come l'Apple, o altri come il VIC 20, il TRS-80, ecc.

Riconosco perfettamente cosa significhi per qualsiasi casa editrice il pubblicare listati di programmi di vari personal, ma spaziando in lungo e in largo ne ho trovati dei più svariati ma sempre per i modelli sopra citati. Ora, arrivando al nocciolo della questione, vi chiedo specificamente per il mio problema (forse molti me ne vorranno per questo) non sarebbe ora di toglierci questi paraocchi ed iniziare ad aiutare anche gli altri utenti di personal? Naturalmente se ho portato avanti questo discorso è perché sono un diretto interessato, possessore di un *TI 99/4A*. Un vero e proprio personal che non dovrebbe essere disprezzato solo per l'appellativo che gli è stato dato di computer da *casa*. Anzi credo proprio che potrebbe essere messo allo stesso livello di molti altri personal, che ora non sto a citare, se non anche su un gradino più in alto. Perché allora non si sono mai visti servizi di nessun genere riguardante questo personal? Dato il mio insuccesso perché non iniziate voi di *Personal Software* a farne qualcuna? Potreste inserire qualche servizio su "I segreti dei

personal".

Rinnovo i miei complimenti alla redazione ed ai collaboratori tutti e colgo l'occasione per augurarvi un buon anno.

Cordiali saluti.

Roberto Marchetti
L'Aquila

Il Texas TI 99/4A è da poco tempo in commercio in Italia, ed inizia solo ora a diffondersi. Un segno di questa diffusione sono le decine di lettere come questa che abbiamo ricevuto.

Naturalmente ne teniamo conto, e ci stiamo attrezzando.

Ma uno dei punti della nostra filosofia è fare una "cultura del software" piuttosto che una distribuzione di "cibi precotti". In questo senso, tutti gli articoli e i programmi che pubblichiamo, anche se non specifici per il Texas, possono adattarsi ad esso. L'esercitarsi in questo campo aiuta a crescere.

Egregio direttore,
ho letto con interesse i primi due numeri della rivista *Personal Software*, ed anche gli editoriali che portano la sua firma. Facendo riferimento all'ultimo capoverso dell'editoriale pubblicato sul n. 2 della rivista, mi permetto di esprimere le seguenti opinioni personali sul sommario e più generalmente sul contenuto della rivista.

1. I programmi finora pubblicati (terza parte del sommario) si riferiscono esclusivamente a giochi.

È pur vero che molti utenti di personal computer utilizzano questo tipo di software, tuttavia questo prodotto è utilizzato da una particolare utenza oppure da utenti diversamente impegnati per i momenti di distensione.

2. Avendo visto la rivista in mano a miei allievi, a colleghi docenti di informatica e ad utenti di personal computer titolari di studi di progettazione, penso che sotto la voce "programmi" possano comparire degnamente anche prodotti software didattico/applicativi. Ad esempio possibili argomenti da pubblicare con la documentazione ed i listati potrebbero essere:

- risoluzione di equazioni non algebriche con metodi iterativi (per studenti, insegnanti e progettisti).
- algoritmi notevoli per la determinazione del massimo o del minimo di una funzione reale di una variabile reale.
- la simulazione mediante elaboratore di fenomeni fisico/chimici, meccanici o idraulici.
- metodi approssimati per il calcolo di derivate, integrali oppure per l'interposizione di punti sperimentali.

Non sapendo al momento quali siano i vostri progetti le invio la documentazione ed il listato della procedura relativa al primo argomento, intendendo che le manderò anche i rimanenti qualora essi presentassero, nello spirito della rivista, un certo interesse.

Con la speranza, ché tale era l'intenzione, di essere stato costruttivo Le invio i più cordiali saluti.

Angelo Cappellini
Pavia

Non è esatto che i programmi pubblicati si riferiscono esclusivamente a giochi. Comunque, come avrà visto, ora la sezione "Programmi" non compare più.

Dopo averla pubblicata per tre numeri, ci siamo resi conto che non aveva ragione di esistere. Un programma richiede sempre un'introduzione, una spiegazione, dei commenti, degli schemi, e tutto ciò trova maggior respiro nello spazio di un articolo.

In questo senso la rivista è fatta di "programmi", e una sezione separata con questo nome ci è sembrata priva di senso.

Per quanto riguarda le sue proposte, le confessiamo che gli argomenti che ci propone ci trovano molto poco entusiasti. Secondo noi si tratta di metodi e tecniche senz'altro utili, ma noiose e in qualche modo "gratuite" se non applicate a qualche problema reale. E poi, insomma, sono cose ormai vecchie, alcune di un paio di secoli, di cui si conosce praticamente tutto, e su cui esistono centinaia di trattati.

Un punto interessante semmai è quello della simulazione, se fatta con un po' di fantasia e... allegria. Aspettiamo contributi in proposito.

Spett.le Direzione,
la presente per rivolgerle una critica, che spero venga utilizzata costruttivamente, sui programmi pubblicati sulla rivista *Personal Software*.

I programmi, relativamente al VIC 20, risultano scritti talmente in piccolo che ne risulta impossibile la lettura e quindi l'utilizzazione, a meno di non usare una forte lente.

I caratteri in campo inverso dei suddetti programmi sono poi assolutamente illeggibili anche usando un ingranditore.

Non è possibile, considerando che l'impiego dei personal si sta estendendo a macchia d'olio, ingrandire i listati in modo da renderli comprensibili ed utilizzabili anche da parte dei principianti?

Non sarebbe possibile raccogliere dopo alcuni numeri della rivista i programmi in una cassetta da allegare alla rivista stessa.

Ringraziando dell'attenzione prestatami siano graditi cordiali saluti.

Bernardino Calza
Piacenza

I suoi rilievi tecnici sono corretti, e sono i nostri problemi nell'impa-

ginare la rivista. Cerchiamo di fare il meglio possibile.

Per quanto riguarda la raccolta su cassetta, pensiamo di farlo dopo aver raggiunto un certo numero di programmi.

Gentili Signori,
sono un ragazzo sedicenne appassionato di informatica e di personal computer, ed ho molto gradito la comparsa della vostra rivista: si sentiva la mancanza di un periodico specializzato nel software.

Certo, vi sono altre riviste dedicate a temi più ampi, quali l'informatica in generale o il mercato dei personal, ma proprio per la loro ampiezza di temi rischiano a volte di diventare inconcludenti.

Pur nella soddisfazione di veder colmata una lacuna in un campo che da fantascienza sta assumendo a mano a mano sempre più la qualifica di "quotidiano", "di massa" (io stesso, fino ad un anno fa non sapevo niente di calcolatori elettronici che non fossero le macchinette da tavolo o da tasca), devo tuttavia notarne una mancanza: nella vostra rivista di programmi, ho notato con rammarico la mancanza dei programmi dedicati al VIC 20/CBM.

Devo anche dire che questa è un'opinione personalissima e faziosa, dato che io personalmente possiedo quel computer, ma in tutta onestà penso che uno sguardo verso i cugini più piccoli lo si potrebbe anche dare (visto che è presente anche la parte del Sinclair).

Senza contare che il VIC ha la possibilità di disegnare in grafica, possibilità che il PET, che pure è della stessa casa produttrice, non ha.

Passando ad altro, devo dire che apprezzo molto la proposta fatta nell'editoriale della "rivista aperta" che accetta articoli di lettori; vorrei inoltre suggerire alcuni temi che potrebbero essere trattati:

- la programmazione dei giochi di scacchiera (mi riferisco soprattutto alla tecnica "di ricerca ad albero" che nell'articolo "Ricorsività in Basic" del n. 1 mi ha molto incuriosito);
- gli altri linguaggi in circolazione sui personal (soprattutto il Pascal);
- perché no? il famigerato linguaggio macchina: "In che cosa si differenzia un programma in LM dal corrispondente in Basic (Pascal, ...)?", "Quali sono i pregi ed i difetti di un programma in LM?", "Come ci si deve porre un problema programmando in LM?" sono dei quesiti che spesso mi pongo.

Sperando che la rivista mantenga tutte le buone qualità che sembra offrire, voglio porvi l'unica domanda: accettereste programmi dei lettori da esaminare, criticare e, a vostra discrezione, pubblicare, su un determinato argomento?

Vi ringrazio per la cortese attenzione nel giungere fino in fondo a questa lettera e vi saluto cortesemente.

Stefano De Santis

Sul VIC 20 abbiamo pubblicato e pubblicheremo. Sulle collaborazioni, l'abbiamo detto: le aspettiamo.

I PUNTI DI FORZA

- **Grafica a 256x192 punti-schermo.**
- **8 colori indipendenti per testo, sfondo, riquadro.**
- **Comandi di suono modulabili in frequenza e durata.**
- **Vera tastiera multifunzione con maiuscole e minuscole. Tutti i tasti con funzione di ripetizione.**
- **Compatibile con teletext.**
- **Alta velocità LOAD e SAVE: 16k byte/ 100 sec.**
- **Funzioni VERIFY e MERGE per programmi e archivi.**
- **BASIC Sinclair esteso con funzioni a 1 tasto; controllo di sintassi.**
- **Ampio software su cassetta.**
- **Perfettamente compatibile con la stampante ZX.**
- **Due modelli:**
  16k byte ROM e 16k byte RAM,
  16k byte ROM e 48k byte RAM.

CPU E MEMORIA ESPANDIBILE
Microprocessore Z80A.
ROM 16k contenente l'interprete BASIC e il sistema operativo.
RAM 16k espandibile a 48k byte.

TASTIERA MULTIFUNZIONE
È dotata di 40 tasti mobili che danno accesso a caratteri maiuscoli e minuscoli ASCII.
Tutte le parole chiave del BASIC sono ottenibili tramite un singolo tasto. Inoltre sono disponibili 16 caratteri grafici, 22 codici di controlli colore e 21 caratteri grafici definibili dall'utente.
Tutti i tasti sono dotati di ripetizione automatica.
Sono presenti i comandi di cursore.

GRAFICA AD ALTA RISOLUZIONE
Lo **ZX Spectrum** può essere collegato direttamente a qualsiasi televisore a colori PAL o in bianconero.
Sono generati 8 colori: nero, blu, rosso, magenta, verde, azzurro, giallo, bianco - sui televisori in bianconero essi appaiono come una regolare scala di grigi.
La grafica è a 256x192 punti. I testi sono visualizzati in 24 linee di 32 caratteri ciascuna. Testo e grafica possono essere sovrapposti. Le istruzioni grafiche BASIC permettono il tracciamento di punti, linee, cerchi ed archi di cerchio.

Di ogni carattere viene memorizzato il colore, il colore dello sfondo, lo stato fisso o lampeggiante, la luminosità normale o extra, il modo diretto o inverso.
Gli attributi di ciascun carattere possono essere determinati indipendentemente da quelli dei caratteri presenti contemporaneamente sullo schermo.
Normalmente le prime 22 righe visualizzano il listato mentre le ultime due sono riservate per evidenziare la linea di programma in fase di editing.
Per l'editing si ricorre ai comandi di cursore.

SUONO
L'altoparlante interno può riprodurre una scala di più di 10 ottave, esattamente 130 semitoni, attraverso il comando BASIC BEEP. Le prese di tipo jack nella parte posteriore del computer permettono la connessione con altoparlanti e amplificatori esterni.

OPERAZIONI E FUNZIONI
Oltre ai normali operatori matematici sono presenti funzioni trascendenti: seno, coseno, tangente e inverse; logaritmi naturali ed esponenziali, funzione segno, valore assoluto, integer, radice quadrata; pigreco; generatore di numeri casuali.
I numeri memorizzati occupano 5 byte: il campo è da 3×10^{-39} a 7×10^{38} con accuratezza di 9½ cifre decimali.
Si possono trattare numeri binari, effettuare operazioni logiche, definire funzioni da parte dell'utente.
È presente un meccanismo completo di DATA, che include i comandi READ, DATA e RESTORE.
Si possono effettuare operazioni sulle stringhe: concatenazione, segmentazione, estrazione di parti.
I vettori possono essere multidimensionali con indici che partono da 1.

ZX Spectrum

16k ÷ 48k byte.
Tastiera multifunzione.
Colore e suono.
Grafica ad alta risoluzione.
Software e hardware ZX già disponibile.
Espandibilità totale.

L. 360.000

più IVA

NELLA VERSIONE 16K RAM

INTERFACCIA CASSETTE

Lo **ZX Spectrum** è dotato di un sofisticato sistema di registrazione su cassette che assicura una registrazione affidabile anche su apparecchi con livello di registrazione automatico.
È possibile registrare su cassetta programmi, interi schermi, blocchi di memoria, vettori contenenti dati.
Programmi e vettori possono essere fusi con altri già esistenti in memoria mediante caricamento dal nastro.
È possibile registrare i programmi in modo da ottenere la partenza automatica del programma nel momento stesso in cui il programma viene ricaricato.
L'interfaccia a cassette opera a 1500 baud tramite 2 jack da 3,5 mm. La velocità è di 16k byte in 100 secondi.

PORTA DI ESPANSIONE

Sul connettore posto nella parte posteriore del computer sono presenti tutte le linee di data address e control propri dello Z80A; tramite questo connettore vengono interfacciate le periferiche.
Sono presenti comandi che permettono di inviare e ricevere dei caratteri da questa porta.

COMPATIBILITÀ CON IL SISTEMA ZX

Il BASIC dello ZX81 è essenzialmente un sottoinsieme del BASIC dello **ZX Spectrum**. Le differenze sono le seguenti: non esistono i comandi FAST e SLOW in quanto lo **ZX Spectrum** opera alla velocità dello ZX81 in maniera FAST avendo comunque una visualizzazione stabile dell'immagine sullo schermo.

Lo **ZX Spectrum** effettua lo SCROLL automaticamente chiedendo all'operatore una conferma ogni volta che lo schermo è pieno.

L'insieme di caratteri dello **ZX Spectrum** è composto da caratteri ASCII al contrario dello ZX81 che adopera un set di caratteri non standard.
I programmi ZX81 possono essere trasferiti sullo **ZX Spectrum** con poche modifiche, e possono essere considerevolmente migliorati grazie alla grafica ed ai colori disponibili.
Le cassette di software registrate con lo ZX81 non possono essere lette dallo **ZX Spectrum**.
Lo **ZX Spectrum** non è compatibile con le espansioni di memoria dello ZX81.
Lo **ZX Spectrum** è pienamente compatibile con la stampante ZX Printer.

è distribuito dalla

REBIT COMPUTER
Via Induno, 18
20092 CINISELLO BALSAMO
Casella Postale 10488 MI

IL BASIC E LA GESTIONE DEI FILE

Il libro si rivolge in modo particolare a chi già conosce il Basic e desidera poter realizzare programmi che prevedano l'uso di file residenti su disco. Dopo aver preso in esame, utilizzando numerosi esempi pratici, le particolarità del Microsoft, si passa alla descrizione delle istruzioni necessarie ad una corretta gestione dei file su disco, sia ad accesso diretto che sequenziale. Una terza parte del libro è infine interamente dedicata alla esposizione dei metodi pratici per l'uso dei file ad accesso diretto e dei data base.

Cod. 515H **L. 11.000 Pagg. 164**

50 ESERCIZI IN BASIC

Una raccolta completa e progressiva di esercizi riguardanti matematica, gestione, ricerca operativa, gioco e statistica. Ciascun esercizio proposto comporta l'enunciazione e l'analisi del problema, la risoluzione mediante flow-chart e commenti, così come un programma che implementa la soluzione, illustrato da semplici esempi rappresentativi.
Questo metodo mette in grado il lettore di verificare passo passo le sue conoscenze e il livello di apprendimento raggiunto.

Cod. 521A **L. 13.000 Pagg. 208**

75 PROGRAMMI IN BASIC PER IL VOSTRO COMPUTER

Il volume raccoglie ben 75 programmi originali scritti in un Basic generico, utilizzabili, salvo poche e semplici modifiche, sulla maggior parte dei personal computer in commercio, a cassetta come a floppy disk. Per ciascuno, dopo una descrizione introduttiva, viene fornito il listing e un campione di esecuzione. Così come sono, i programmi proposti (tutti verificati), costituiscono un valido ausilio per chiunque debba risolvere problemi di matematica, statistica, finanza o, genericamente, di pratica utilità.

Cod. 551D **L. 12.000 Pagg. 196**

GIOCARE IN BASIC

Nei giochi, il lettore può ritrovare tutte quelle situazioni reali di programmazione che gli saranno indispensabili nella comprensione e realizzazione di qualsiasi applicazione interattiva del proprio computer, anche le più sofisticate. Questo senza annoiarsi, ma entrando da subito all'interno della materia per imparare a comprendere il BASIC, il proprio computer e i computer in genere.

Cod. 522A **L. 20.000 Pagg. 324**

SCONTO 10% agli abbonati

CEDOLA DI COMMISSIONE LIBRARIA

Ritagliare (o fotocopiare) e inviare a
Gruppo Editoriale Jackson
Via Rosellini, 12 - 20124 Milano

Nome e Cognome ______________________

____

Indirizzo ______________________

____

Cap. ________ Città ______________________

Provincia ______________________

Partita I.V.A. (indispensabile per le aziende)

| | | | | | | | | | | |

☐ Si richiede l'emissione della fattura

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

☐ Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione

☐ Allego assegno n° .... di L. ....

Data .... Firma

☐ Non Abbonato ☐ Abbonato sconto 10% ☐ L. Elettronica ☐ Elettronica Oggi ☐ Automazione Oggi ☐ Elektor ☐ Informatica Oggi ☐ Computerworld ☐ Bit ☐ Personal Software ☐ Strumenti Musicali ☐ Videogiochi

Interi in precisione multipla

Con queste routine il vostro computer potrà eseguire operazioni aritmetiche tra numeri interi con qualunque numero di cifre

di *Antonio Filz*

Le routine descritte in questo articolo hanno un gran numero di applicazioni.

Tra queste, il calcolo dei fattoriali, la fattorizzazione in numeri primi, e, recentemente, la criptografia.

Ma per i lettori di Personal Software *può essere interessante andare a vedere a pag. 17 del primo numero. Lì si trattava della "congettura di Utam" e le poche cifre dei numeri interi non permettevano di esplorare a fondo il problema. Ora, con queste routine, invece... [M.B.]*

C'è una cosa dei personal computer (ma d'altra parte anche di quasi tutti gli altri calcolatori) che mi ha sempre dato un certo fastidio. Hanno una precisione limitata: generalmente 9 cifre significative. (In questo articolo farò riferimento a Apple II, che ho utilizzato per la stesura dei programmi, e che ha appunto una precisione di 9 cifre significative. Le routine che seguono hanno però una validità del tutto generale, come si vedrà più avanti). Facendo riferimento agli interi, che sono gli unici numeri che prenderemo in considerazione, ciò vuol dire che i numeri utilizzabili sono soltanto quelli con meno di dieci cifre (compresi tra -10^9+1 e 10^9-1). Se vengono superati questi limiti, il computer passa automaticamente alla notazione esponenziale e non si ottengono più dei risultati esatti, ma soltanto loro approssimazioni.

In molti casi sarebbe utile poter effettuare i calcoli con una precisione maggiore. Consideriamo ad esempio il problema di calcolare il fattoriale di un numero N (N!). Il fattoriale di un numero intero non negativo è così definito:

$$N! = \begin{cases} 1 & \text{se } N=0 \\ N\cdot(N-1)\cdot\ldots\cdot 2\cdot 1 & \text{se } N>0 \end{cases}$$

A parte il caso $N=0$, il fattoriale di N non è altro che il prodotto dei primi N numeri interi. Molti problemi di combinatoria implicano la valutazione del fattoriale di numeri più o meno grandi. Per esempio: in quanti modi possono essere disposte le 52 carte di un mazzo? La risposta è 52! Questo numero ha 68 cifre. In effetti, già quando $N=13$, N! ha più di 9 cifre. Come fare quindi a calcolare questi numeri enormi? Per superare questa fastidiosa limitazione ho scritto delle routine che permettono di sommare, sottrarre, moltiplicare, dividere (divisione con resto), elevare a potenza, confrontare, inserire, visualizzare, ecc., numeri interi arbitrariamente grandi. La lunghezza di questi numeri è limitata soltanto dalla memoria del vostro personal e dai tempi di elaborazione.

È importante chiarire che queste routine da sole non fanno nulla. Chi le utilizza deve scrivere un proprio programma principale che le consideri subroutine e le richiami mediante istruzioni GOSUB. Per comprendere bene questo fatto esaminate attentamente l'esempio riportato più avanti.

Vediamo ora com'è organizzato questo package.

Descrizione delle routine

20000 Inizializzazione Questa routine, che deve essere chiamata all'inizio del vostro programma principale, serve a preparare, in

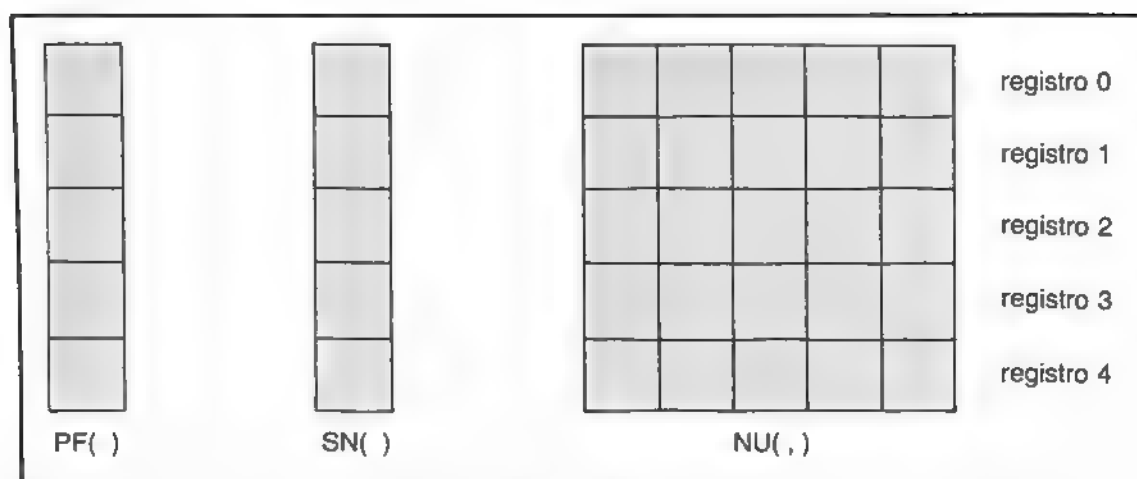



Figura 1. *Esempio di organizzazione di cinque registri.*

base alle vostre esigenze, un certo numero di registri che conterranno i numeri "grandi". (D'ora in poi chiamerò numeri grandi quelli contenuti in questi registri, e numeri piccoli quelli delle variabili usuali). I registri sono costituiti dalla matrice NU (,), che conterrà le cifre dei numeri, e dai vettori SN( ) e PF( ) che conterranno rispettivamente i segni dei numeri (−1 negativo, 0 nullo, 1 positivo) ed i puntatori alla fine dei numeri. Ogni registro è costituito da una riga della matrice NU(,) e da un elemento di SN( ) e uno di PF( ). Nella figura 1 è riportato, ad esempio, come verrebbero organizzati 5 registri.

La routine 20000 fissa inoltre il numero massimo di cifre che possono essere contenute in un elemento di NU(,) (memorizzato in CF) e la base di rappresentazione dei numeri (BB). Infatti i numeri grandi vengono considerati come numeri in base BB ed ogni elemento di NU(,) contiene "una cifra" di questi numeri in base BB.

Al momento dell'esecuzione vi sarà chiesto quante sono le cifre significative del vostro personal, quanti registri volete avere a disposizione e quante cifre al massimo può avere un numero grande. In base alle vostre risposte la routine dimensiona la matrice ed i vettori. Ricordate che se risponderete di volere *K* registri, i loro numeri saranno 0, 1,..., *K*−1. Alcune routine utilizzano dei registri per memorizzare risultati intermedi, quindi se pensate di utilizzarle tenetene conto. La tavola 1 riporta una lista delle routine che utilizzano dei registri e quali registri usano.

Ecco in un caso specifico cosa apparirà sullo schermo quando sarà eseguita la routine 20000.

```
CIFRE SIGNIFICATIVE: 9
REGISTRI:            5
CIFRE PER NUMERO:    20
```

A questo punto avrete a disposizione 5 registri che potranno contenere numeri interi con non più di 20 cifre. Se in un momento qualsiasi si tentasse di utilizzare un numero con più cifre, apparirà il messaggio NUMERO TROPPO GRANDE e l'esecuzione si fermerà. In effetti può capitare di riuscire a lavorare anche con numeri un po' più lunghi, ma questo dipende dalla precisione del vostro personal e da quante cifre può avere ogni numero grande.

Vediamo con un paio di esempi come saranno rappresentati i numeri in questi registri loro riservati. Consideriamo il numero 1234567890987654321. Esso verrà spezzato, a partire dal fondo, in gruppi di 4 cifre (in generale verrà spezzato in gruppi di INT(CS/2) cifre, dove CS sono le cifre significative del vostro personal), e memorizzato come indicato nella figura 2.

Il numero −1408177496532 sarà invece memorizzato come indicato nella figura 3.

Per memorizzare lo 0 non occorre far altro che azzerare il valore del segno.

Forse questo modo di memorizzare numeri vi sembrerà piuttosto complesso (perché non usare semplicemente una stringa?), ma vi assicuro che è efficiente. Quelli tra voi che studieranno a fondo le routine qui riportate, se ne renderanno ben presto conto.

Figura 2. *Memorizzazione del numero 1234567890987654321.*

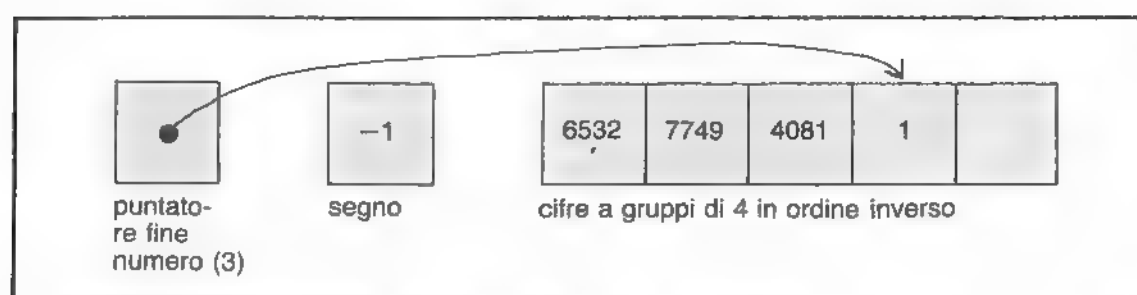



Figura 3. *Memorizzazione del numero −1408177496532.*

16000, 17000 Numeri→registri Queste due routine permettono di memorizzare un numero in un certo registro. La 16000 sistema automaticamente un numero, contenuto sotto forma di stringa nella variabile alfanumerica C$, nel registro C. Supponiamo di voler mettere il numero 1234567890987654321

ROUTINE	SCOPO	REGISTRI UTILIZZATI
8000	Addizione, sottrazione o moltiplicazione	$K-1$
9000	Elevamento a potenza	$K-1,...,K-3$
9500	Elevamento a potenza	$K-1,...,K-4$
10000	Divisione con resto	$K-1,...,K-5$

Tavola 1. *Registri utilizzati da alcune routine (supponendo che ci siano* K *registri, da 0 a* K−1*).*

nel registro 0. Non dobbiamo far altro che scrivere queste tre istruzioni (precedute naturalmente dai loro rispettivi numeri di linea):

C$="12345678909876543 21" (non mettere il segno "+" davanti ai numeri positivi)

C=0
GOSUB 16000

Per memorizzare −140817749 6532 nel registro 2 basta scrivere:

C$="−1408177496532"
C=2
GOSUB 16000

In questo modo è quindi possibile usare anche istruzioni DATA che contengono numeri grandi sotto forma di stringhe, leggere questi numeri mediante istruzioni READ e memorizzarli nei registri desiderati.

La routine 17000 memorizza il numero piccolo NT nel registro C. Ad esempio, per mettere il numero 2 nel registro 1, si può scrivere:

NT=2
C=1
GOSUB 17000

Questa routine è stata scritta supponendo che la funzione STR$ *non* metta uno spazio davanti a numeri positivi o nulli. Se il vostro personal mette questo spazio, cambiate le linee 17010-17040 con:

```
17010 IF NT>0 THEN SN(C)=1:
      GOTO 17030
17020 SN(C)=−1
17030 A$=STR$(NT)
17040 A$=MID$(A$,2)
```

5000, 6000 Addizione, sottrazione Servono per eseguire rispettivamente l'addizione e la sottrazione dei numeri contenuti nei registri A e B. Il risultato è messo nel registro C. Ad esempio, per effettuare la somma di due numeri memorizzati nei registri 1 e 3 e mettere il risultato nel registro 0, basta scrivere:

A=1
B=3
C=0
GOSUB 5000

Se invece volessimo eseguire la sottrazione, basterebbe sostituire GOSUB 5000 con GOSUB 6000.

È interessante capire come sono strutturate queste due routine, che sono intimamente legate fra di loro. Non è difficile osservare che la somma e la differenza di due numeri possono essere calcolate in base agli schemi riportati nella tavola 2. Notate che in qualsiasi caso bisogna calcolare la somma o la differenza dei valori assoluti dei due numeri. Poi, se il primo numero è negativo, occorre cambiare segno al risultato. Il calcolo della somma e della differenza dei valori assoluti è eseguito rispettivamente dalle subroutine 5100 e 6100, che agiscono essenzialmente nello stesso modo in cui si fanno i calcoli

SOMMA		SECONDO NUMERO	
		NEGATIVO	POSITIVO O NULLO
PRIMO NUMERO	NEGATIVO	SOMMA = − (SOMMA VALORI ASSOLUTI)	SOMMA = DIFFERENZA VALORI ASSOLUTI
	POSITIVO O NULLO	SOMMA = − (DIFFERENZA VALORI ASSOLUTI)	SOMMA = SOMMA VALORI ASSOLUTI

DIFFERENZA		SECONDO NUMERO	
		NEGATIVO	POSITIVO O NULLO
PRIMO NUMERO	NEGATIVO	DIFFERENZA = − (DIFFERENZA VALORI ASSOLUTI)	DIFFERENZA = − (SOMMA VALORI ASSOLUTI)
	POSITIVO O NULLO	DIFFERENZA = SOMMA VALORI ASSOLUTI	DIFFERENZA = DIFFERENZA VALORI ASSOLUTI

Tavola 2. *Relazione tra somma e differenza di due numeri e somma e differenza tra i valori assoluti dei due numeri.*

manualmente. Partendo dalle cifre meno significative si effettuano i calcoli tenendo conto degli eventuali riporti o prestiti.

Due interessanti formule per il massimo e il minimo di due numeri

C'è un'istruzione interessante, la 5210. Serve per trovare il minimo tra IA e IB. Il minimo tra due numeri n_1 e n_2 è dato da:

$$\min(n_1,n_2) = (n_1+n_2-|n_1-n_2|)/2$$

dove $|n_1-n_2|$ indica il valore assoluto della differenza tra n_1 e n_2.

Una formula analoga per il massimo tra due numeri è:

$$\max(n_1,n_2) = (n_1+n_2+|n_1-n_2|)/2$$

7000 Moltiplicazione Effettua la moltiplicazione tra i contenuti dei registri A e B. Il risultato viene memorizzato nel registro C. Ad esempio, se in un certo programma si volesse valutare il prodotto di due numeri che si trovano nei registri 4 e 5, e mettere il risultato nel registro 1, basterebbero queste quattro istruzioni:

```
A=4
B=5
C=1
GOSUB 7000
```

Anche questa routine effettua il calcolo in modo molto simile al procedimento manuale. L'unica differenza consiste in questo. Manualmente, una moltiplicazione consiste di due fasi distinte. Nella prima si sviluppa il calcolo moltiplicando ogni cifra del secondo numero per il primo numero, con relativo shift a sinistra. Nellฺa seconda si fa la somma di tutte le righe dello sviluppo, ottenendo il risultato finale.

Nella routine 7000 esiste un unico processo che effettua contemporaneamente moltiplicazioni e somme parziali.

8000 Operazioni con numeri piccoli Permette di effettuare operazioni tra un numero grande ed uno piccolo; il risultato è sempre un numero grande. In base al valore di SO (selettore di operazione), che può essere 1, 2 o 3, esegue rispettivamente l'addizione, la sottrazione o la moltiplicazione tra il contenuto del registro A ed il numero piccolo NP e mette il risultato nel registro C. Questa routine non fa altro che memorizzare il valore di NP nel registro $K-1$ utilizzando la routine 7000, e poi chiamare una delle routine precedentemente descritte (5000, 6000 o 7000).

9000, 9500 Elevamento a potenza La routine 9000, per quanto riguarda l'algoritmo usato, è sicuramente la più interessante di tutto il package. Essa eleva il contenuto del registo A alla potenza ES (ES è una

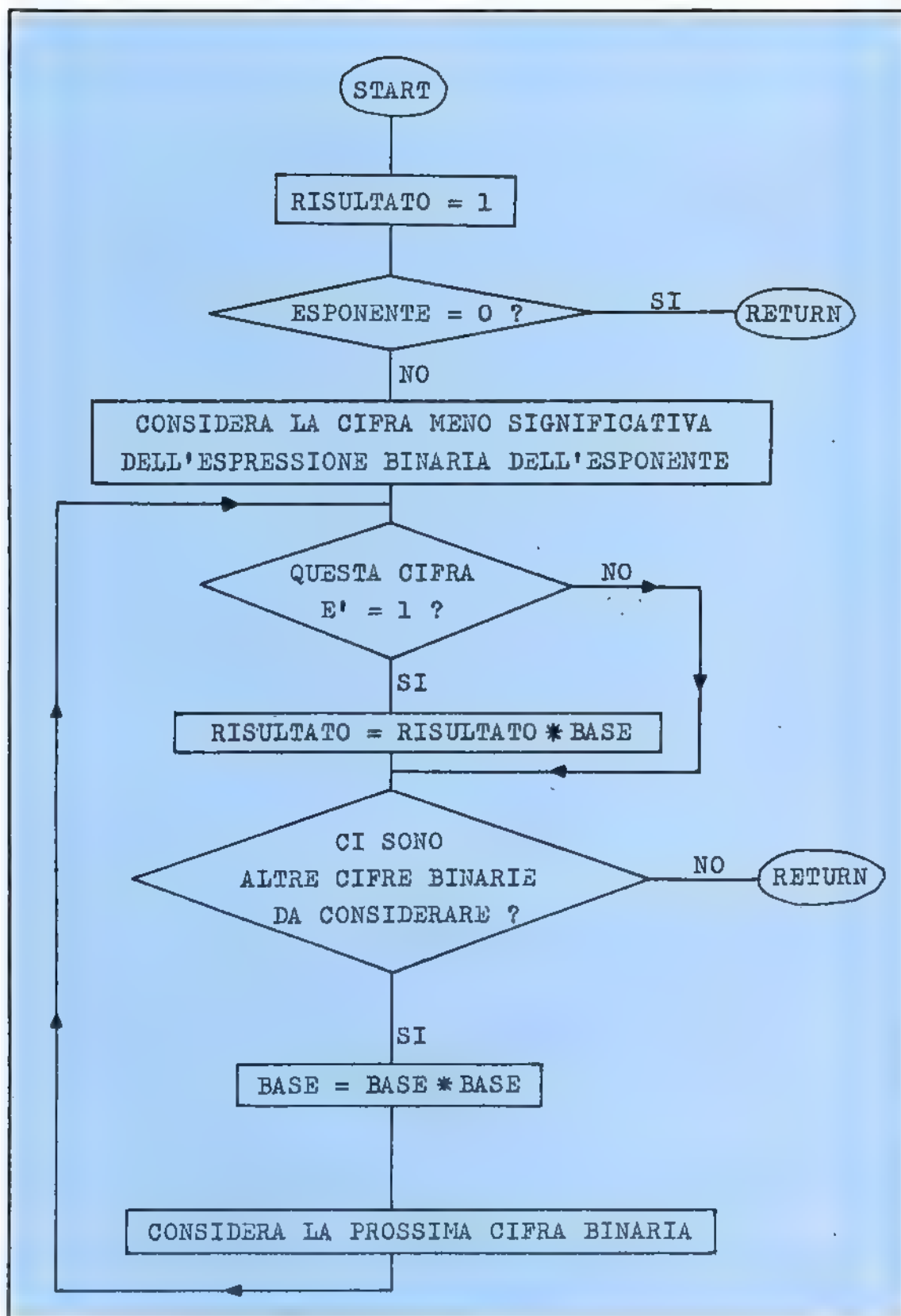



Tavola 3. *Diagramma di flusso dell'algoritmo di elevamento a potenza.*

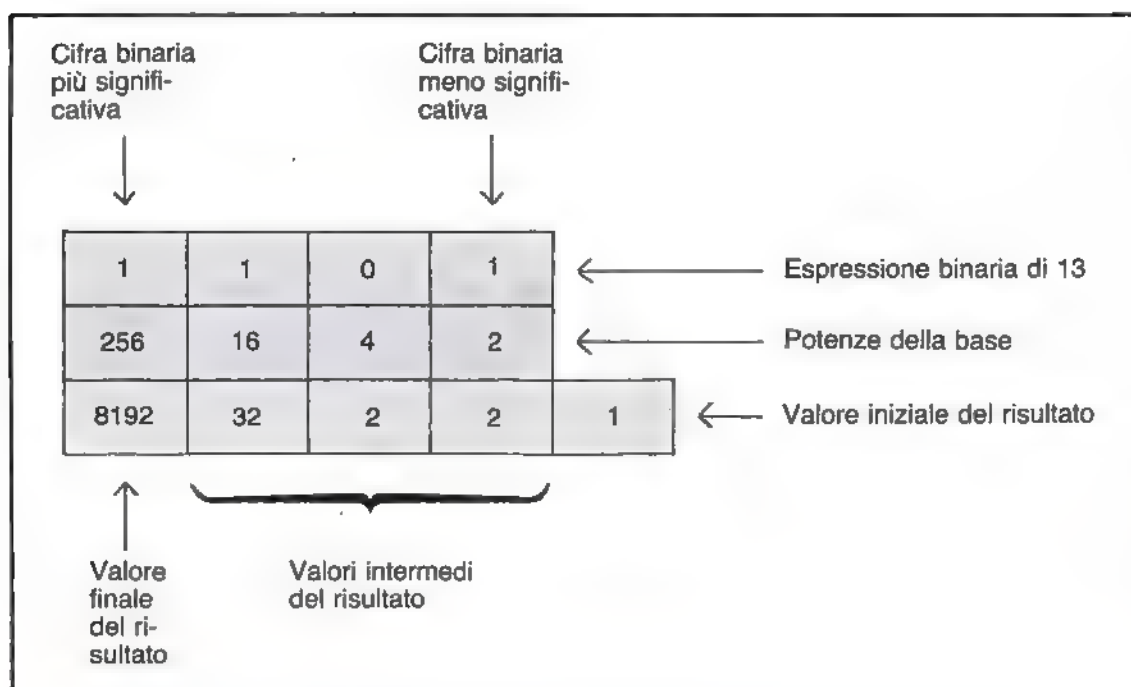



Figura 4. *Schema per l'elevamento a potenza.*

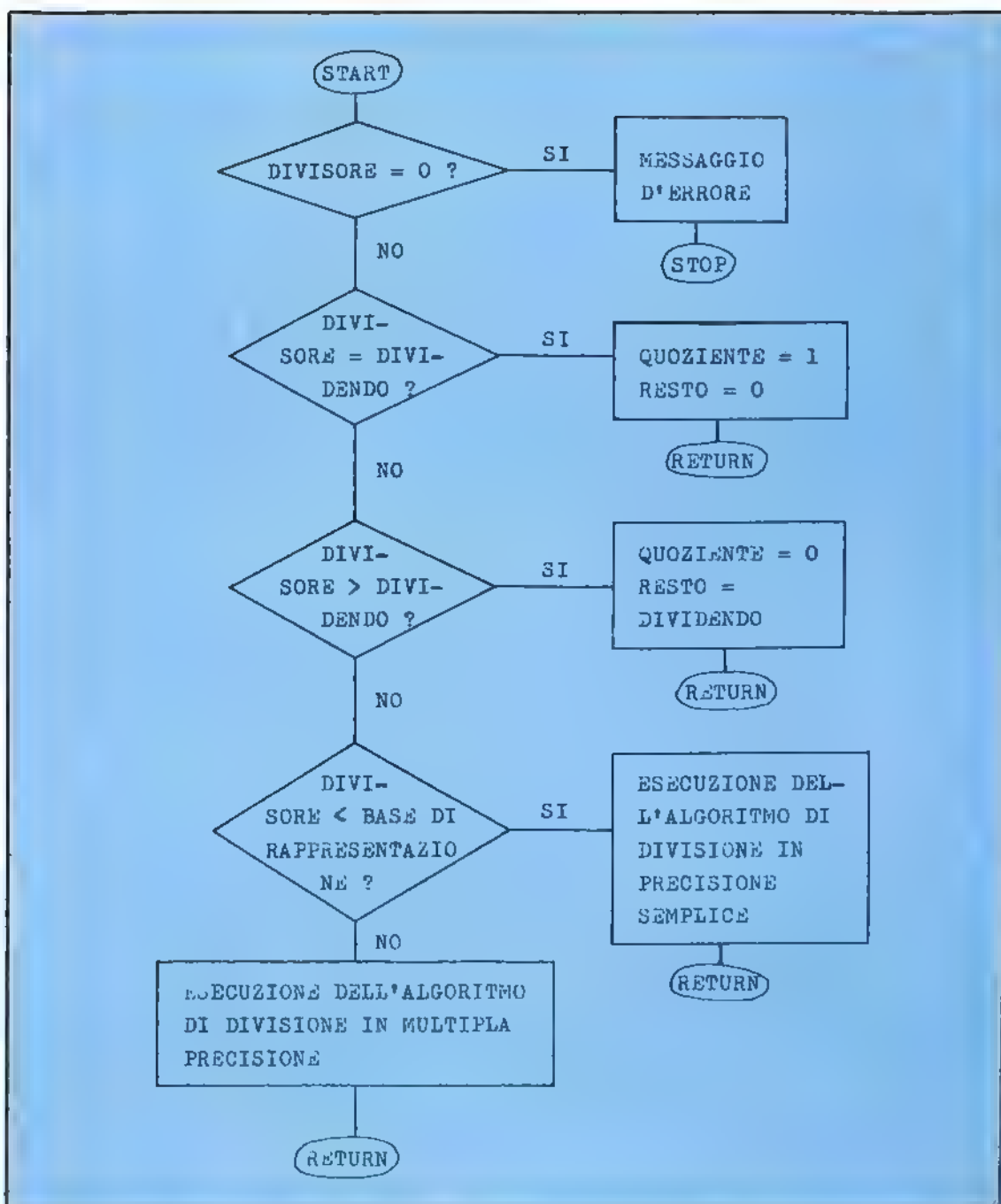



Tavola 4. *Struttura della routine di divisione con resto.*

variabile usuale che deve contenere un valore intero non negativo), e mette il risultato nel registro C.

Il metodo più semplice per eseguire un elevamento a potenza consiste nel moltiplicare la base per se stessa un numero di volte uguale all'esponente meno uno. Per esempio, per valutare 2^{13}, occorrerebbero 12 moltiplicazioni. Questo metodo va bene soltanto quando l'esponente è piccolo (minore di 10).

Nella routine 9000 è stato invece utilizzato un sistema molto più efficace, detto metodo binario per l'esponenziazione. Il diagramma di flusso dell'algoritmo (vedi tavola 3), può spiegare meglio di un lungo discorso il funzionamento di questa routine. Per specificarlo meglio, vediamo come opera questo metodo nella valutazione di 2^{13}. Lo schema di figura 4 riassume tutte le fasi del calcolo.

In parole povere si pensa a 2^{13} come a $2 \cdot 2^4 \cdot 2^8$, cioè al prodotto di potenze della base che hanno come esponenti potenze di 2 distinte.

Per non perdere il valore della base, nei calcoli viene utilizzata una copia di quest'ultima.

Notate che questa routine fornisce 1 come valore di 0^0, il che non ha molto senso, poiché 0^0 è definibile soltanto come limite.

La routine 9500 permette di elevare il numero piccolo BS alla potenza ES. Il contenuto di BS viene memorizzato nel registro $K-4$ con una chiamata alla routine 17000, e poi si esegue l'elevamento a potenza utilizzando la routine 9000.

1000 Divisione con resto Questa routine, che effettua la divisione con resto tra i valori assoluti di due numeri, è la più complessa di tutto il package. Permette di valutare quoziente e resto della divisione tra i valori assoluti dei numeri contenuti nei registri A e B. Il quoziente viene memorizzato nel registro C ed il resto nel registro D. Nel caso in cui il divisore valga zero viene stampato il messaggio TENTATIVO DI DIVISIONE PER ZERO e l'esecuzione si arre-

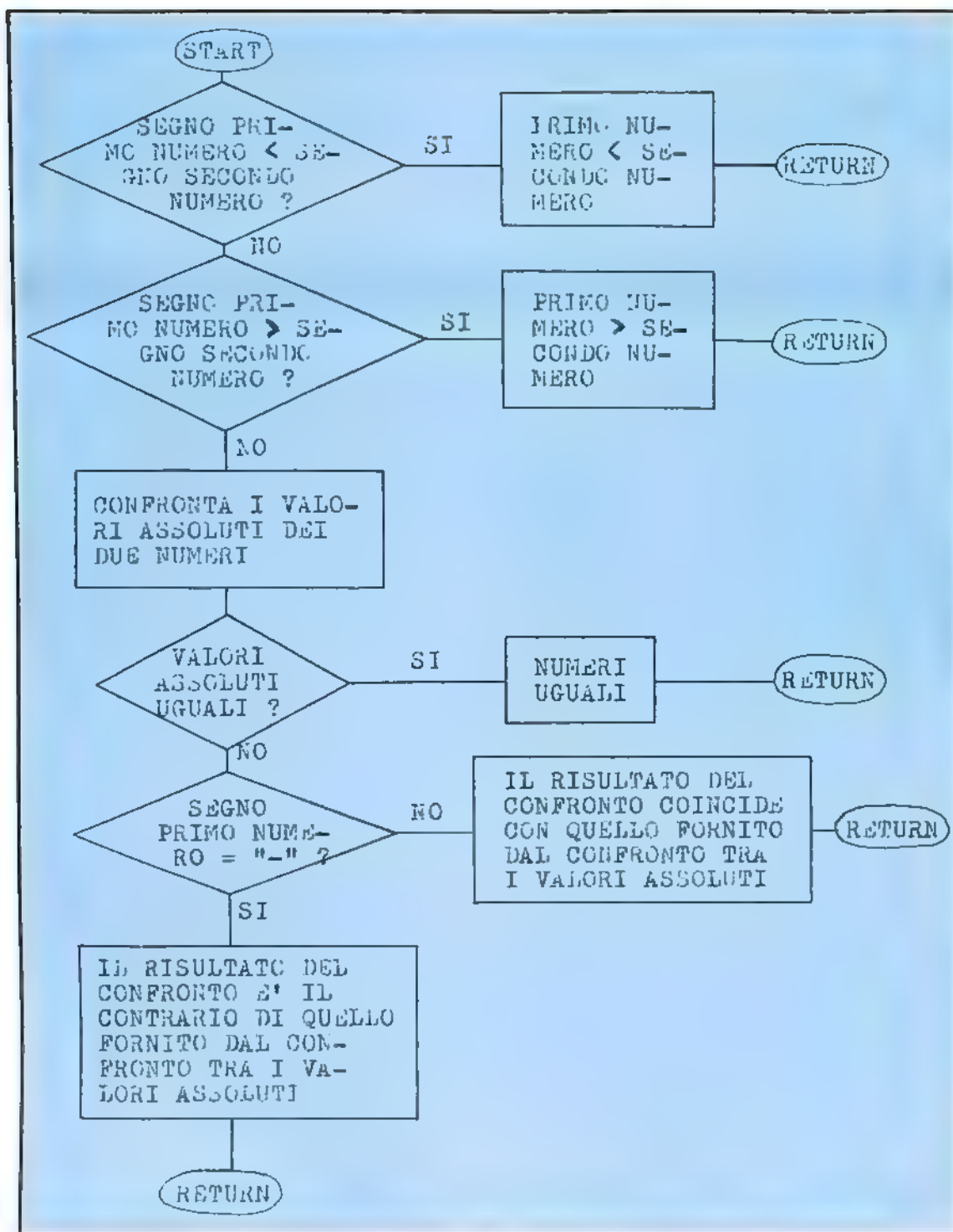



Tavola 5. *Diagramma di flusso.*

sta. Altrimenti vengono individuati alcuni casi particolari in cui si valutano quoziente e resto molto facilmente. Nei casi più complessi il calcolo è eseguito praticamente nello stesso modo in cui si fa manualmente e tutti i dettagli possono essere trovati nel libro citato alla fine dell'articolo. La struttura di questa routine è riportata nel diagramma di flusso della tavola 4.

Supponiamo di voler valutare la divisione con resto tra i valori assoluti dei numeri contenuti nei registri 0 e 1, e di voler mettere quoziente e resto rispettivamente nei registi 3 e 4. Dovremo scrivere le seguenti istruzioni:

```
A=0
B=1
C=3
GOSUB 10000
```

14000 Confronto È la routine che permette di confrontare due numeri grandi contenuti nei registri A e B. In uscita si ha:

$$CO = \begin{cases} -1 \text{ se numero nel registro A} \\ \quad < \text{numero nel registro B} \\ 0 \text{ se numero nel registro A} \\ \quad = \text{numero nel registro B} \\ 1 \text{ se numero nel registro A} \\ \quad > \text{numero nel registro B} \end{cases}$$

Quindi, per sapere il risultato del confronto, dopo l'istruzione GOSUB 14000 bisogna controllare il valore della variabile CO.

Questa routine utilizza la subroutine 14100 (richiamata anche dalla 6100) per il confronto dei valori assoluti dei numeri.

Le tavole 5 e 6 riportano rispettivamente i diagrammi di flusso delle routine 14000 e 14100.

13000 Visualizzazione Visualizza il numero contenuto nel registro C. Il numero zero è considerato come un caso a parte. Altrimenti, se il numero è negativo viene visualizzato il segno –. Poi, partendo dalle cifre più significative, vengono stampati gli elementi della matrice NU(,) che costituiscono il numero, aggiungendo, se necessario, degli zeri. Alla fine viene eseguita l'istruzione PRINT, in modo che la successiva stampa inizierà su una nuova riga. Per visualizzare il numero contenuto nel registro 2, occorre scrivere:

```
C=2
GOSUB 13000
```

Anche per questa routine si suppone che la funzione STR$ *non* ponga uno spazio davanti a numeri positivi o nulli. Se il vostro personal lo mette, aggiungete la seguente linea:

```
13045 A$=MID$(A$,2)
```

15000 Copia Copia il contenuto del registro A nel registro C. Ecco le tre istruzioni necessarie per copiare il contenuto del registro 2 nel registro 1:

```
A=2
C=1
GOSUB 15000
```

18000 Azzera Azzera completamente il registro C. Viene richiamata dalla routine 7000.

Un'applicazione: la valutazione di polinomi

Come esempio concreto di utilizzazione di questo package, ho scel-

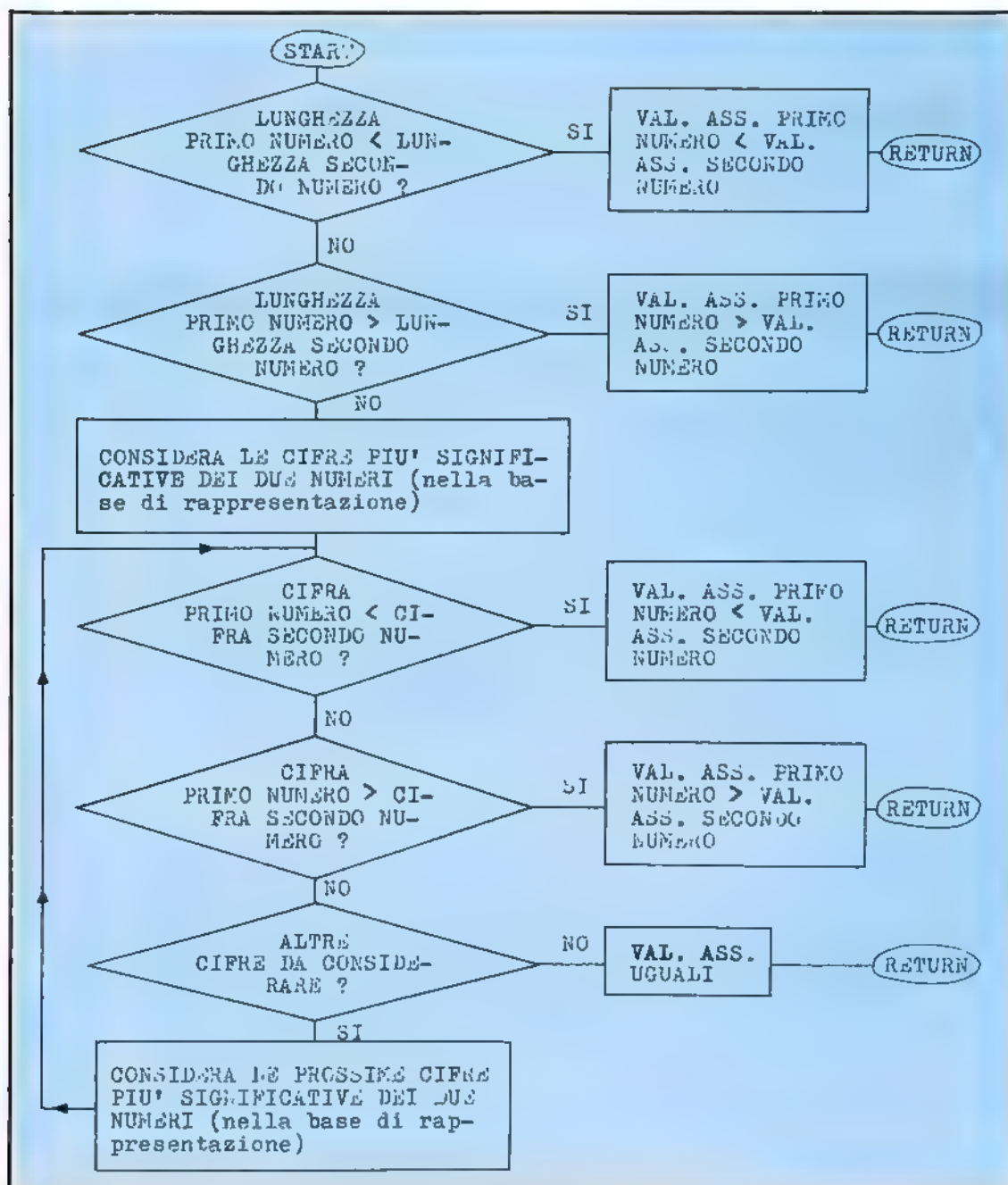



Tavola 6. *Diagramma di flusso della subroutine di confronto tra valori assoluti.*

to il problema della valutazione di un polinomio. Si tratta di calcolare il valore assunto da un polinomio della forma

$$P(X)=a_nX^n+a_{n-1}X^{n-1}+\ldots$$
$$+a_1X+a_0 \quad a_n\neq 0$$

per un certo valore della variabile X.

Conviene riscrivere il polinomio nel seguente modo

$$P(X)=(\ldots(a_nX+a_{n-1})X+a_{n-2})X+$$
$$+\ldots)X+a_0$$

ed effettuare il calcolo mediante un ciclo che esegue ogni volta la moltiplicazione per X e la somma di un nuovo coefficiente a partire dalla parentesi più interna.

Sia i coefficienti a_i che X devono essere numeri piccoli.

Analizziamo il programma.

La linea 20 dimensiona il vettore dei coefficienti.

Le linee 40-70 servono per la preparazione di 4 registri con una capacità di 40 cifre l'uno. Se il vostro personal ha un numero di cifre significative diverso da 9 cambiate la linea

```
40 CS=9
```

con una appropriata. Avrete notato che c'è GOSUB 20030, anziché GOSUB 20000. Questo serve per evitare le istruzioni di input della routine di inizializzazione. Poi (linee 80-180) vi vengono richiesti alcuni dati: il grado del polinomio, cioè l'esponente della massima potenza di X, che deve essere ≤ 20, i coefficienti del polinomio, ed infine il valore di X.

Se l'istruzione STR$ del vostro personal mette uno spazio davanti ai numeri positivi o nulli, cambiate la linea 130 con

```
130 I$=MID$(STR$(I),2)
```

La chiamata alla subroutine 600 (riga 190) causa la visualizzazione del polinomio. Questa routine è stata scritta appositamente per Apple II. Chi non possedesse un Apple II la può ignorare e cambiare la linea 190 con

```
190 PRINT: PRINT"VALORE DEL POLINOMIO IN ";X;" =";
```

Con questo programma potete calcolare il valore di un polinomio intero in un punto

Naturalmente in questo caso l'output sarà molto meno elegante.

La subroutine 600 è piuttosto interessante. Essa effettua la stampa del polinomio, su più righe se necessario, in una forma matematicamente corretta. Le tre parti principali della routine sono quelle che fissano, per ogni termine del polinomio, il segno del coefficiente, le cifre del coefficiente e l'esponente di X. Per quanto riguarda l'esponente, esso non verrà stampato per gli ultimi due termini, in quanto l'ultimo termine non contiene nemmeno la variabile e nel penultimo l'esponente sarebbe 1 e allora si tralascia.

Il segno viene sempre scritto, tranne nel caso del primo termine con coefficiente positivo.

Le cifre dei coefficienti vengono sempre scritte quando i coefficienti *non* valgono 1 o −1. In questi ultimi due casi viene scritto 1 soltanto se stiamo trattando l'ultimo termine, che non moltiplica X.

La valutazione vera e propria del polinomio inizia alla linea 200.

Se il grado del polinomio è zero si passa direttamente alla visualiz-

zazione. Altrimenti (linee 250-300) il valore di X e quello del coefficiente a_n sono memorizzati rispettivamente nei registri 0 e 3.

A questo punto viene eseguito il ciclo compreso tra le linee 310 e 430, che sarà percorso un numero di volte uguale al grado del polinomio. Ad ogni passaggio, il risultato parziale viene moltiplicato per X e a questo è sommato un nuovo coefficiente.

La visualizzazione del risultato è effettuata chiamando la routine 13000 (linea 440).

Poi apparirà un piccolo menù con tre possibilità:

1 – Valutare lo stesso polinomio per un altro valore di X.
2 – Cambiare polinomio.
3 – Uscire dal programma.

La stampa dell'esempio di esecuzione riportato nella tavola 7 è stata ottenuta utilizzando una routine di utilità in linguaggio macchina che effettua la stampa di tutto quello che appare sul video.

Bibliografia e ringraziamenti

A chi volesse approfondire l'argomento, consiglio il libro:

D.E. Knuth *The Art of Computer Programming, vol. 2: Seminumerical Algorithms* (second edition). Addison-Wesley, 1980.

In particolare i paragrafi
4.3.1 The Classical Algorithms
4.6.3. Evaluation of Powers
4.6.4. Evaluation of Polynomials

È una lettura molto impegnativa, ma altrettanto interessante.

Desidero ringraziare la Software House SIGE di Trento, che mi ha gentilmente messo a disposizione un Apple II completo di floppy disk driver e stampante, nonché un Apple III in simulazione di Apple II quando quest'ultimo non era disponibile. ■

Questo programma è disponibile su disco per l'Apple. Vedete nelle ultime pagine il "Servizio programmi".

Variabili utilizzate e loro uso

VARIABILE/I	DESCRIZIONE
NU (,)	Matrice contenente le cifre dei numeri grandi.
SN ( )	Vettore dei segni dei numeri.
PF ( )	Vettore dei puntatori alla fine dei numeri.
CS	Cifre significative del vostro personal.
K	Numero di registri con cui si lavora.
N	La prima volta, massimo numero di cifre di un numero grande; poi, indice dell'ultima colonna della matrice NU (,).
CF	Numero massimo di cifre contenute in ogni elemento della matrice NU (,).
BB	Base di rappresentazione dei numeri grandi (10^{CF}).
A, B, C, D	Indicatori di registri.
SO	Selettore di operazione.
NP	Numero piccolo con cui eseguire un'operazione con un numero grande.
ES	Esponente per le routine di elevamento a potenza.
BS	Base nella routine di elevamento a potenza 9500.
CO	Risultato del confronto tra due numeri grandi.
NT	Numero piccolo da memorizzare in un registro.
C$	Stringa numerica da memorizzare in un registro.
I, IA, IB, IC, ID	Indici di ciclo.
A$, CB, CE, CT, D1, DB, DS, LU, TT, Z	Variabili di lavoro.
A1, A2, A3, B1, B2, B3, C1, C2, C3	Copie dei valori di A, B, C.

Nota: I valori delle variabili A, B, C, D, SO, NP, ES, BS, CO, NT, C$ vengono mantenuti inalterati da ogni routine.

```
GRADO DEL POLINOMIO = 15

COEFFICIENTI DEL POLINOMIO

A(15) = 12
A(14) = 0
A(13) = -1
A(12) = 234
A(11) = 1
A(10) = 544
A(9) = 323
A(8) = 32
A(7) = 2
A(6) = -24
A(5) = -21
A(4) = 0
A(3) = 0
A(2) = -234
A(1) = 5567
A(0) = 1

VALORE DI X = 4


          15   13        12   11        10       9
P(X)=12X   -X   +234X   +X   +544X   +323X  +

        8    7     6     5      2
    -32X +2X -24X -21X -234X -5567X+1


P(4)=17404982365


1 - ALTRO VALORE DI X

2 - NUOVO POLINOMIO

3 - FINE
```

Tavola 7.

Quadro riassuntivo delle routine

ROUTINE	SCOPO	DESCRIZIONE
5000	Addizione	Esegue l'addizione tra i contenuti dei registri A e B, e mette il risultato nel registro C.
6000	Sottrazione	Esegue la sottrazione tra i contenuti dei registri A e B, e mette il risultato nel registro C.
7000	Moltiplicazione	Esegue la moltiplicazione tra i contenuti dei registri A e B, e mette il risultato nel registro C.
8000	Addizione, Sottrazione o Moltiplicazione	Esegue, in base al valore di SO (1, 2 o 3), l'addizione, la sottrazione o la moltiplicazione tra il contenuto del registro A ed il numero piccolo NP. Mette il risultato nel registro C.
9000	Elevamento a potenza	Eleva il contenuto del registro A alla potenza ES (numero piccolo), e mette il risultato nel registro C.
9500	Elevamento a potenza	Eleva il numero piccolo BS alla potenza ES (numero piccolo) e mette il risultato nel registro C.
10000	Divisioni con resto	Valuta la divisione tra i contenuti dei registri A e B. Quoziente nel registro C e resto nel registro D.
13000	Visualizzazione	Visualizza il contenuto del registro C.
14000	Confronto	Confronta i contenuti dei registri A e B. In uscita si ha CO = −1, CO = 0 oppure CO = 1 a seconda che il contenuto del registro A sia minore, uguale o maggiore di quello del registro B.
15000	Copia	Copia il contenuto del registro A nel registro C.
16000	Stringa in un registro	Memorizza la stringa numerica C$ nel registro C.
17000	Numero piccolo in un registro	Memorizza il numero piccolo NT nel registro C.
18000	Azzera un registro	Azzera il registro C.
20000	Inizializzazione	Dimensiona la matrice NU (,) ed i vettori SN( ) e PF( ) in base alle esigenze dell'utilizzatore. Fissa inoltre la base di rappresentazione.

Nota: Non è ammessa nessuna coincidenza tra i valori di A, B, e C al momento della chiamata alle routine 5000, 6000, 7000 e 8000, tra quelli di A, B, C e D per quanto riguarda la routine 10000, e tra quelli di A e C per la routine 9000.

Listato 1. *Interi in precisione multipla.*

```
4999  REM ADDIZIONE
5000  IF SN(A) THEN 5060
5010  A1=A
5020  A=B
5030  GOSUB 15000
5040  A=A1
5050  RETURN
5060  IF SN(B)<>0 THEN 5090
5070  GOSUB 15000
5080  RETURN
5090  CO=2
5100  Z=ABS(SN(A)-SN(B))/2+1
5110  ON Z GOSUB 5200,6200
5120  IF SN(A)<>-1 OR CO=0 THEN RETURN
5130  SN(C)=-SN(C)
5140  RETURN
5199  REM ADDIZIONE VALORI ASSOLUTI
5200  SN(C)=1
5210  LU=(PF(A)+PF(B)-ABS(PF(A)-PF(B)))/2
5220  GOSUB 5300
5230  IF PF(A)=PF(B) THEN 5270
5240  IF PF(A)>PF(B) THEN Z=A: GOTO 5260
5250  Z=B
5260  GOSUB 5400
5270  GOSUB 5500
5280  RETURN
5300  CB=0
5310  FOR I=0 TO LU
5320  NU(C,I)=NU(A,I)+NU(B,I)+CB
5330  IF NU(C,I)<BB THEN CB=0: GOTO 5360
5340  NU(C,I)=NU(C,I)-BB
5350  CB=1
5360  NEXT I
5370  RETURN
5400  FOR I=LU+1 TO PF(Z)
5410  NU(C,I)=NU(Z,I)+CB
5420  IF NU(C,I)<BB THEN CB=0: GOTO 5450
5430  NU(C,I)=NU(C,I)-BB
5440  CB=1
5450  NEXT I
5460  RETURN
5500  IF CB=0 THEN PF(C)=I-1: RETURN
5510  IF I=N THEN PRINT"NUMERO TROPPO GRANDE":
      STOP
5520  NU(C,I)=1
5530  PF(C)=I
5540  RETURN
5999  REM SOTTRAZIONE
6000  IF SN(A)<>0 THEN 6070
6010  A1=A
6020  A=B
6030  GOSUB 15000
6040  SN(C)=-SN(C)
6050  A=A1
6060  RETURN
6070  IF SN(B)<>0 THEN 6100
6080  GOSUB 15000
6090  RETURN
6100  CO=2
6105  CO=2
6110  Z=ABS(SN(A)-SN(B))/2+1
6120  ON Z GOSUB 6200,5200
6130  IF SN(A)<>-1 OR CO=0 THEN [illegible]
6140  SN(C)=-SN(C)
6150  RETURN
6199  REM SOTTRAZIONE VALORI [illegible]
6200  GOSUB 14100
6210  IF CO<>0 THEN 6240
6220  SN(C)=0
6230  RETURN
6240  IF CO=-1 THEN 62[illegible]
6250  GOSUB 6400
6260  RETURN
6270  A1=A
6280  A=B
6290  B=A1
6300  GOSUB 6400
6310  SN(C)=-1
6320  B=A
6330  A=A1
6340  RETURN
6400  SN(C)=1
6410  GOSUB 6500
6420  IF PF(A)=PF(B) THEN [illegible]
```

(segue)

Listato 1. *Interi in precisione multipla (segue).*

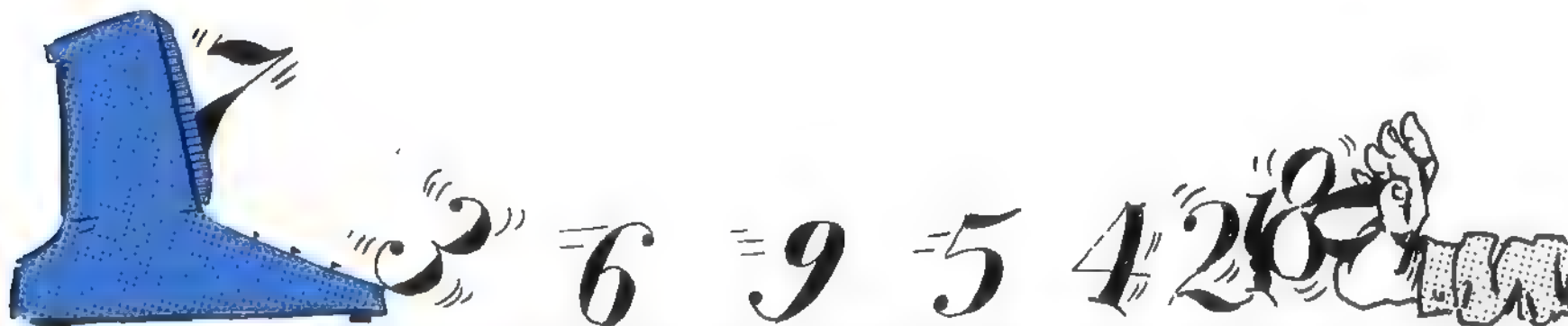
```
6430  GOSUB 6600
6440  GOSUB 6700
6450  RETURN
6500  CB=0
6510  FOR I=0 TO PF(B)
6520  NU(C,I)=NU(A,I)-NU(B,I)-CB
6530  IF NU(C,I)>=0 THEN CB=0: GOTO 6560
6540  NU(C,I)=NU(C,I)+BB
6550  CB=1
6560  NEXT I
6570  RETURN
6600  FOR I=PF(B)+1 TO PF(A)
6610  NU(C,I)=NU(A,I)-CB
6620  IF NU(C,I)>=0 THEN CB=0: GOTO 6650
6630  NU(C,I)=NU(C,I)+BB
6640  CB=1
6650  NEXT I
6660  RETURN
6700  FOR I=PF(A) TO 0 STEP-1
6710  IF NU(C,I)<>0 THEN 6730
6720  NEXT I
6730  PF(C)=[illegible]
6740  RETURN
6999  REM MOLTIPLICAZIONE
7000  IF SN(A)<>0 AND SN(B)<>0 THEN 7030
7010  SN(C)=0
7020  RETURN
7030  GOSUB [illegible]
7040  FOR IA=[illegible]
7050  FOR IB=0 TO PF(B)
7060  IC=IA+IB
7070  IF IC=N THEN PRINT"NUMERO TROPPO GRANDE":
      STOP
7080  NU(C,IC)=NU(C,IC)+NU(A,IA)*NU(B,IB)
7090  NEXT IB
7100  FOR I=IA TO IC
7110  IF NU(C,I)<BB THEN 7150
7120  IF I=N-1 THEN PRINT"NUMERO TROPPO GRANDE":
      STOP
7130  Z=INT(NU(C,I)/BB)
7140  NU(C,I)=NU(C,I)-Z*BB
7150  NU(C,I+1)=NU(C,I+1)+Z
7160  NEXT I
7170  NEXT IA
7180  IF I=N THEN PF(C)=I-1: GOTO 7210
7190  IF NU(C,I)=0 THEN PF(C)=I-1: GOTO 7210
7200  PF(C)=I
7210  SN(C)=SN(A)*SN(B)
7220  RETURN
7998  REM ADD. SOTTR. O MOLT.
7999  REM TRA GRANDE E PICCOLO
8000  C2=C
8010  C=K-1
8020  C1=NT
8030  NT=NP
8040  GOSUB 17000
8050  B2=B
8060  B=C
8070  C=C2
8080  ON SO GOSUB 5000,6000,7000
8090  B=B2
8100  NT=C1
8110  RETURN
8998  REM ELEVAMENTO A POTENZA
8999  REM BASE=GRANDE ESP.=PICCOLO
9000  SN(C)=1
9010  PF(C)=0
9020  NU(C,0)=1
9030  IF ES=0 THEN RETURN
9040  A1=A
9050  B1=B
9060  C1=C
9070  C=[illegible]
9080  GOSUB 15000
9090  CE=ES
9100  CB=CE-INT(CE/2)*2
9110  IF CB=0 THEN 9190
9120  A=C1
9130  C=K-1
9140  GOSUB 15000
9150  A=K-3
9160  B=C
9170  C=C1
9180  GOSUB 7000
9190  CE=(CE-CB)/2
9200  IF CE=0 THEN 9290
9205  A=K-3
9210  C=K-2
9220  GOSUB 15000
9230  C=K-1
9240  GOSUB 15000
9250  A=K-1
9255  B=K-2
9260  C=K-3
9270  GOSUB 7000
9280  GOTO 9100
9290  A=A1
9300  B=B1
9310  C=C1
9320  RETURN
9498  REM ELEVAMENTO A POTENZA
9499  REM BASE=PICCOLO ESP.=PICCOLO
9500  NT=BS
9510  C1=C
9520  C=K-4
9530  GOSUB 17000
9540  A2=A
9550  A=K-4
9560  C=C1
9570  GOSUB 9000
9580  A=A2
9590  RETURN
9998  REM DIVISIONE CON RESTO TRA VALORI
9999  REM ASSOLUTI DI NUMERI GRANDI
10000 IF SN(B)=0 THEN PRINT
      "TENTATIVO DI DIVISIONE PER ZERO": STOP
10010 GOSUB 14100
10020 IF CO<>0 THEN 10080
10030 SN(C)=1
10040 PF(C)=0
10050 NU(C,0)=1
10060 SN(D)=0
10070 RETURN
10080 IF CO=1 THEN 10160
10090 SN(C)=0
10100 C1=C
10110 C=D
10120 GOSUB 15000
10130 SN(D)=ABS(SN(D))
10140 C=C1
10150 RETURN
10160 SN(C)=1
10170 SN(D)=1
10180 IF PF(B)<>0 THEN 10300
10190 PF(D)=0
10200 NU(D,0)=0
10210 FOR I=PF(A) TO 0 STEP-1
10220 CB=NU(D,0)*BB+NU(A,I)
10230 NU(C,I)=INT(CB/NU(B,0))
10240 NU(D,0)=CB-NU(C,I)*NU(B,0)
10250 NEXT I
10260 IF NU(C,PF(A))=0 THEN PF(C)=PF(A)-1:
      RETURN
10270 PF(C)=PF(A)
10280 RETURN
10300 A3=A
10310 B3=B
10320 C3=C
10330 TT=NP
10340 D1=INT(BB/(NU(B,PF(B))+1))
10350 NP=D1
10360 C=K-3
10370 SO=3
10380 GOSUB 8000
10390 IF PF(K-3)=PF(A) THEN PF(K-3)=PF(K-3)+1:
      NU(K-3,PF(K-3))=0
10400 A=B3
10410 C=K-2
10420 GOSUB 8000
10430 PF(K-4)=PF(B)+1
```

(segue)

Listato 1. *Interi in precisione multipla (segue).*

```
10440 FOR ID=PF(K-3) TO PF(K-2)+1 STEP-1
10450 IF NU(K-3,ID)=NU(K-2,PF(K-2)) THEN DB=BB-1:
      GOTO 10470
10460 DB=INT((NU(K-3,ID)*BB+NU(K-3,ID-1))/NU(K-2,
      PF(K-2)))
10470 DS=(NU(K-3,ID)*BB+NU(K-3,ID-1)-DB*NU(K-2,
      PF(K-2)))*BB+NU(K-3,ID-2)
10480 IF NU(K-2,PF(K-2)-1)*DB>DS THEN DB=DB-1:
      GOTO 10470
10490 SN(K-4)=0
10500 FOR I=ID TO ID-PF(K-2)-1 STEP-1
10510 IF NU(K-3,I)<>0 THEN SN(K-4)=1
10520 NU(K-4,PF(K-2)+1-ID+I)=NU(K-3,I)
10530 NEXT I
10540 NP=DB
10550 A=K-2
10560 C=K-5
10570 GOSUB 8000
10580 A=K-4
10590 B=K-5
10600 C=K-1
10610 GOSUB 6000
10640 IF PF(C)=PF(A) THEN 10680
10650 FOR I=PF(C)+1 TO PF(K-4)
10660 NU(C,I)=0
10670 NEXT I
10680 LU=0
10690 IF SN(C)=-1 THEN 10770
10700 SN(C)=1
10710 FOR I=0 TO PF(A)
10720 NU(C,I)=BB-NU(C,I)-LU
10730 LU=1
10740 NEXT I
10770 FOR I=ID TO ID-PF(A) STEP-1
10780 NU(K-3,I)=NU(C,PF(A)-ID+I)
10790 NEXT I
10800 NU(C3,ID-PF(A))=DB
10810 IF LU=0 THEN 10910
10820 NU(C3,ID-PF(A))=DB-1
10830 LU=0
10840 FOR I=ID-PF(A) TO ID-1
10850 NU(K-3,I)=NU(K-2,I-ID+PF(A))+LU
10860 IF NU(K-3,I)<BB THEN LU=0: GOTO 10890
10870 NU(K-3,I)=NU(K-3,I)-BB
10880 LU=1
10890 NEXT I
10900 NU(K-3,I)=NU(K-3,I)+LU-BB
10910 NEXT ID
10920 PF(C3)=PF(K-3)-PF(K-2)-1
10930 IF NU(C3,PF(C3))=0 THEN PF(C3)=PF(C3)-1
10940 LU=0
10950 FOR I=PF(K-2) TO 0 STEP-1
10960 CE=LU*BB+NU(K-2,I)
10970 NU(D,I)=INT(CE/D1)
10980 LU=CE-NU(D,I)*D1
10990 NEXT I
11000 SN(D)=1
11010 PF(D)=PF(K-2)
11020 FOR I=PF(D) TO 0 STEP-1
11030 IF NU(D,I)<>0 THEN PF(D)=I: GOTO 1106[illegible]
11040 NEXT I
11050 SN(D)=0
11060 A=A3
11070 B=B3
11080 C=C3
11090 NP=T1
11100 RETURN
12999 REM VISUALIZZAZIONE
13000 IF SN(C)=0 THEN PRINT"0": RETURN
13010 IF SN(C)=-1 THEN PRINT"-";
13020 Z=0
13030 FOR I=PF(C) TO 0 STEP-1
13040 A$=STR$(NU(C,I))
13050 IF Z=0 THEN 13110
13060 LU=LEN(A$)
13070 IF LU=CF THEN 13110
13080 FOR IC=1 TO CF-LU
13090 A$="0"+A$
13100 NEXT IC
13110 PRINT A$;
13120 Z=1
13130 NEXT I
13140 PRINT""
13150 RETURN
13999 REM CONFRONTO
14000 IF SN(A)<SN(B) THEN CO=-1: RETURN
14010 IF SN(A)>SN(B) THEN CO=1: RETURN
14020 GOSUB 14100
14030 IF SN(A)=-1 THEN CO=-CO
14040 RETURN
14099 REM CONFRONTO VALORI ASSOLUTI
14100 IF PF(A)<PF(B) THEN CO=-1: RETURN
14110 IF PF(A)>PF(B) THEN CO=1: RETURN
14120 FOR I=PF(A) TO 0 STEP-1
14130 IF NU(A,I)<NU(B,I) THEN CO=-1: RETURN
14140 IF NU(A,I)>NU(B,I) THEN CO=1: RETURN
14150 NEXT I
14160 CO=0
14170 RETURN
14999 REM COPIA DI UN NUMERO
15000 SN(C)=SN(A)
15010 PF(C)=PF(A)
15020 FOR I=0 TO PF(A)
15030 NU(C,I)=NU(A,I)
15040 NEXT I
15050 RETURN
15999 REM STRINGA NUMERICA - REGISTRO
16000 IF C$="0" THEN SN(C)=0: RETURN
16010 A$=C$
16020 IF LEFT$(A$,1)<>"-" THEN SN(C)=1:
      GOTO 160[illegible]
16030 SN(C)=-1
16040 A$=MID$(A$,2)
16050 LU=LEN(A$)
16060 IF LU>N*CF THEN PRINT"NUMERO TROPPO GRANDE":
      STOP
16070 PF(C)=INT((LU-1)/CF)
16080 FOR I=PF(C) TO 0 STEP-1
16090 IF LU>CF THEN 16120
16100 NU(C,PF(C)-I)=VAL(A$)
16110 RETURN
16120 NU(C,PF(C)-I)=VAL(RIGHT$(A$,CF))
16130 A$=LEFT$(A$,LU-CF)
16140 LU=LEN(A$)
16150 NEXT I
16999 REM NUMERO PICCOLO-- REGISTRO
17000 IF NT=0 THEN SN(C)=0: RETURN
17010 A$=STR$(NT)
17020 IF NT>0 THEN SN(C)=1: GOTO 170[illegible]
17030 SN(C)=-1
17040 A$=MID$(A$,2)
17050 FOR I=0 TO 2
17060 LU=LEN(A$)
17070 IF LU<=CF THEN NU(C,I)=VAL(A$): PF(C)=I:
      RETURN
17080 NU(C,I)=VAL(RIGHT$(A$,CF))
17090 A$=LEFT$(A$,LU-CF)
17100 NEXT I
17999 REM AZZERA UN NUMERO
18000 SN(C)=0
18010 PF(C)=0
18020 FOR I=0 TO N-1
18030 NU(C,I)=0
18040 NEXT I
18050 RETURN
19999 REM INIZIALIZZAZIONE
20000 INPUT"CIFRE SIGNIFICATIVE  : [illegible]
20010 INPUT"REGISTRI             : [illegible]
20020 INPUT"CIFRE PER NUMERO     : [illegible]
20030 CF=INT(CS/2)
20040 BB=INT(10^CF+0.5)
20050 N=INT((N-1)/CF)+1
20060 DIM NU(K-1,N),SN(K-1),PF(K-1)
20070 RETURN
```

Listato 2. *Valutazione di polinomi.*

```
10      REM VALUTAZIONE DI POLINOMI
20      DIM A(20)
30      HOME
40      CS=9
50      K=4
60      N=40
70      GOSUB 20030
80      INPUT"GRADO DEL POLINOMIO = ";G
90      PRINT
100     PRINT"COEFFICIENTI DEL POLINOMIO"
110     PRINT
120     FOR I=G TO 0 STEP-1
130     I$=STR$(I)
140     PRINT"A(";I$;") = ";
150     INPUT"";A(I)
160     NEXT I
170     PRINT
180     INPUT"VALORE DI X = ";X
190     GOSUB 600
200     IF G>0 THEN 250
210     NT=A(0)
220     C=0
230     GOSUB 17000
240     GOTO 440
250     NT=X
260     C=3
270     GOSUB 17000
280     NT=A(G)
290     C=0
300     GOSUB 17000
310     FOR L=G-1 TO 0 STEP-1
320     A=3
330     B=0
340     C=1
350     GOSUB 7000
360     NT=A(L)
370     C=2
380     GOSUB 17000
390     A=1
400     B=2
410     C=0
420     GOSUB 5000
430     NEXT L
440     GOSUB 13000
450     PRINT: PRINT
460     PRINT"1 - ALTRO VALORE DI X": PRINT
470     PRINT"2 - NUOVO POLINOMIO": PRINT
480     PRINT"3 - FINE": PRINT
490     GET L$
500     IF L$<>"1" AND L$<>"2" AND L$<>"3" THEN 490
510     HOME
520     L=VAL(L$)
530     ON L GOTO 180,80,540
540     END
599     REM STAMPA POLINOMIO
600     HOME
610     VTAB(3)
620     PRINT"P(X)=";
630     PS=5: RS=3
635     NN=0
640     FOR L=G TO 0 STEP-1
650     B$="": E$=""
660     IF A(L)=0 THEN 940
665     NN=1
670     LN=0
680     IF A(L)<0 THEN B$=B$+"-": LN=1: GOTO 720
690     IF L=G THEN 720
700     B$=B$+"+"
710     LN=1
720     IF ABS(A(L))<>1 THEN Z$=STR$(ABS(A(L))):
        B$=B$+Z$: LN=LN+LEN(Z$): GOTO 760
730     IF L<>0 THEN 760
740     B$=B$+"1"
750     LN=LN+1
760     IF L=0 THEN 820
770     B$=B$+"X"
780     LN=LN+1
790     IF L=1 THEN 820
800     E$=STR$(L)
810     LN=LN+LEN(E$)
820     IF PS+LN<39 THEN 880
830     IF A(L)<0 THEN PRINT"-": GOTO 850
840     PRINT"+"
850     PRINT: PRINT
855     RS=RS+3
860     PRINT SPC(5)
870     PS=5
880     PRINT B$;
890     VTAB(RS-1)
910     PRINT E$;
915     VTAB(6)
920     VTAB(RS)
930     PS=PS+LN
940     NEXT L
945     IF NN=0 THEN PRINT"0";
950     PRINT"": PRINT: PRINT
960     Z$=STR$(X)
970     PRINT"P(";Z$;")=";
980     RETURN
```

3

Il controllo del codice fiscale

Mauro Boscarol

La routine Basic di questo mese controlla l'esattezza di un codice fiscale.

Secondo la legislazione italiana, il codice fiscale è composto di 16 caratteri alfanumerici. I primi sei caratteri sono alfabetici e codificano il cognome e nome. I prossimi cinque caratteri codificano la data di nascita e il sesso: i primi due sono numerici, e indicano le ultime due cifre dell'anno, il terzo è alfabetico e indica il mese secondo la corrispondenza riportata in tabella 1, e gli ultimi due sono numerici e indicano il giorno per i soggetti maschili, oppure il giorno + 40 per i soggetti femminili.

Seguono poi un carattere alfabetico e tre caratteri numerici che codificano il luogo di nascita. Infine, il sedicesimo è un carattere alfabetico di controllo.

(A = alfabetico, N = numerico)

Per esempio, il mio codice fiscale è

BSCMRA47B10A952O

I primi sei caratteri codificano il mio cognome e nome. Gli altri cinque la mia data di nascita e il sesso. I prossimi quattro il luogo di nascita (Bolzano), e l'ultimo è il carattere di controllo: una O maiuscola, non uno zero.

gennaio	A	luglio	L
febbraio	B	agosto	M
marzo	C	settembre	P
aprile	D	ottobre	R
maggio	E	novembre	S
giugno	H	dicembre	T

Tabella 1. *La codifica del mese di nascita*

I controlli da effettuare

Quando si è in possesso di un codice fiscale, per controllare se è stato trascritto correttamente, si possono effettuare i seguenti controlli:

(a) il codice deve essere lungo 16 caratteri
(b) i primi sei devono essere alfabetici (A-Z)
(c) i caratteri di posto 7 e 8 devono essere numerici (00-99)
(d) il carattere di posto 9 deve essere alfabetico (vedi tabella 1)
(e) i caratteri di posto 10 e 11 devono essere numerici (01-31 o 41-71)
(f) il carattere di posto 12 deve essere alfabetico
(g) i caratteri di posto 13, 14 e 15 devono essere numerici
(h) il carattere di controllo deve essere alfabetico ed inoltre...

Il sedicesimo carattere

L'articolo 7 della legge sul codice fiscale (Gazzetta Ufficiale n. 345 del 29 dicembre 1976) specifica che il sedicesimo carattere del codice ha funzione di controllo della esatta trascrizione dei primi quindici caratteri.

A o 0	1	N	20
B o 1	0	O	11
C o 2	5	P	3
D o 3	7	Q	6
E o 4	9	R	8
F o 5	13	S	12
G o 6	15	T	14
H o 7	17	U	16
I o 8	19	V	10
J o 9	21	W	22
K	2	X	25
L	4	Y	24
M	18	Z	23

Tabella 2. *Conversioni per i caratteri di posto dispari*

A o 0	0	N	13
B o 1	1	O	14
C o 2	2	P	15
D o 3	3	Q	16
E o 4	4	R	17
F o 5	5	S	18
G o 6	6	T	19
H o 7	7	U	20
I o 8	8	V	21
J o 9	9	W	22
K	10	X	23
L	11	Y	24
M	12	Z	25

Tabella 3. *Conversioni per i caratteri di posto pari*

Esso viene determinato (a partire dai primi quindici) con il seguente algoritmo:

1 Ad ognuno degli otto caratteri di posto dispari assegnare un valore secondo la tabella 2
2 Ad ognuno dei sette caratteri di posto pari assegnare un valore secondo la tabella 3
3 Sommare i 15 valori così ottenuti e dividere (con resto) il risultato per 26
4 Convertire il resto ottenuto nel carattere alfabetico ad esso corrispondente secondo la tabella 4. Questo carattere è il carattere di controllo.

Per esempio, controllo il mio codice fiscale. Inizio con il convertire i caratteri di posto dispari usando la tabella 2

B	S	C	M	R	A	4	7	B	1	0	A	9	5	2	
0		5		8		9		0		1		21		5	

Poi converto quelli di posto pari mediante la tabella 3

B	S	C	M	R	A	4	7	B	1	0	A	9	5	2	
0	18	5	12	8	0	9	7	0	1	1	0	21	5	5	

0	A	13	N
1	B	14	O
2	C	15	P
3	D	16	Q
4	E	17	R
5	F	18	S
6	G	19	T
7	H	20	U
8	I	21	V
9	J	22	W
10	K	23	X
11	L	24	Y
12	M	25	Z

Tabella 4. *Conversione del resto nel carattere di controllo*

Sommo tutti i valori ottenuti e ho 92, che diviso per 26 dà 3 con resto 14: il carattere di controllo che ricavo dalla tabella 4 è quindi O.

Il programma

Nelle righe 20-90 vengono definiti mediante DATA il vettore C(16) che contiene 0 se il corrispondente carattere deve essere alfabetico e 1 se deve essere numerico e il vettore D(26) che corrisponde alla tabella 2.

Nelle righe 100-130 c'è un esempio di chiamata della routine, che inizia alla riga 10000 (nelle righe 9940-9990 vi sono le condizioni di ingresso e uscita).

```
10    DIM C(16),D(26)
20    DATA 0,0,0,0,0,0,1,1,0,1,1,0,1,1,1,0
30    DATA 1,0,5,7,9,13,15,17,19,21,2,4,18,
      20,11,3,6,8,12,14,16,10,22,25,24,23
40    FOR I=1 TO 16
50    READ C(I)
60    NEXT I
70    FOR I=1 TO 26
80    READ D(I)
90    NEXT I
100   INPUT A$
110   GOSUB 10000
120   IF Z=1 THEN PRINT"ERRATO": STOP
130   PRINT"GIUSTO": STOP
9940  REM
9950  REM CONTROLLO DEL CODICE FISCALE
9960  REM
9970  REM VARIABILE IN INPUT:
      A$(CODICE FISCALE)
9980  REM VARIABILE IN OUTPUT:
      Z(0=CORRETTO,1=ERRATO)
9990  REM
10000 Z=1
10010 IF LEN(A$)<>16 THEN RETURN
10020 I=1
10030 B$=MID$(A$,I,1)
10040 IF B$<"A" OR B$>"Z" GOTO 10070
10050 IF C(I)<>0 THEN RETURN
10060 GOTO 10110
10070 IF B$<"0" OR B$>"9" THEN RETURN
10080 IF C(I)<>1 THEN RETURN
10090 I=I+1
10100 IF I<=16 GOTO 10030
10110 REM
10120 REM CARATTERE DI CONTROLLO
10130 REM
10140 S=0
10150 FOR I=1 TO 15 STEP 2
10160 B$=MID$(A$,I,1)
10170 IF C(I)=0 GOTO 10200
10180 S=S+D(VAL(B$)+1)
10190 GOTO 10210
10200 S=S+D(ASC(B$)-64)
10210 IF I=15 GOTO 10270
10220 B$=MID$(A$,I+1,1)
10230 IF C(I+1)=0 GOTO 10260
10240 S=S+VAL(B$)
10250 GOTO 10270
10260 S=S+ASC(B$)-65
10270 NEXT I
10280 R=S-INT(S/26)*26
10290 IF MID$(A$,16,1)<>CHR$(R+65) THEN
      RETURN
10300 Z=0
10310 RETURN
```

IL TUO PRIMO COMPUTER

sinclair ZX81

Il computer più venduto nel mondo

lo trovi anche nel tuo "bit shop primavera"

ALESSANDRIA Via Savonarola, 13
ANCONA Via De Gasperi, 40
AREZZO Via F. Lippi, 13
BARI Via Devitofrancesco, 4/2A
BARI Via Capruzzi, 192
BARLETTA Via Vitrani, 58
BASSANO DEL GRAPPA Via Jacopo Da Ponte, 51
BERGAMO Via S. F. D'Assisi, 5
BIELLA Via Italia, 50A
BOLOGNA Via Brugnoli, 1
CAGLIARI Via Zagabria, 47
CAMPOBASSO Via Mons II Bologna, 10
CESANO MADERNO Via Ferrini, 6
CINISELLO BALSAMO V.le Matteotti, 66
COMO Via L. Sacco, 3
COSENZA Via Dei Mille, 86
CUNEO C.so Nizza, 16
FAVRIA CANAVESE C so G Matteotti, 13
FIRENZE Via G. Milanesi, 28/30
FOGGIA Via Marchianò, 1
FORLÌ P.zza Melozzo Degli Ambrogi, 1
GALLARATE Via A. Da Brescia, 2
GENOVA Via Domenico Fiasella, 51/R
GENOVA-SESTRI Via Chiaravagna, 10/R
IMPERIA Via Delbecchi, 32
L'AQUILA Strada 85 N 2
LECCO Via L. Da Vinci, 7
LIVORNO Via San Simone, 31
LUCCA Via S Concordio, 160
MACERATA Via Spalato, 126
MERANO Via S Maria del Conforto, 22
MESSINA Via Del Vespro, 71
MILANO Via G Cantoni, 7
MILANO Via E. Petrella, 6
MILANO Via Altaguardia, 2
MILANO P.zza Firenze, 4
MILANO V.le Corsica, 14
MILANO V.le Certosa, 91
MILANO Via Jacopo Palma, 9
MONZA Via Azzone Visconti, 39
MORBEGNO Via Fabani, 31
NAPOLI Via Luigio Sanfelice, 7/A
NAPOLI C so Vittorio Emanuele, 54
NOVARA Baluardo Q Sella, 32
PADOVA Via Fistomba, 8
PALERMO Via Libertà, 191
PARMA Via Imbriani, 41
PAVIA Via C Battisti, 4/A
PERUGIA Via R. D'Andreotto, 49/55
PESCARA Via Tiburtina, 264 bis
PESCARA Via Trieste, 73
PIACENZA Via IV Novembre, 60
PISA Via XXIV Maggio, 101
PISTOIA V.le Adua, 350
POTENZA Via G Mazzini, 72
POZZUOLI Via G.B Pergolesi, 13
RIMINI Via Bertola, 75
ROMA L.go Belloni, 4 (Vigna Stelluti)
ROMA P zza San Donà Di Piave, 14
ROMA V.le IV Venti, 152
ROMA Via Cerreto Da Spoleto, 23
SAVONA Via G Scarpa, 13B
SONDRIO Via N Sauro, 28
TERAMO Via Martiri Pennesi, 14
TERNI Via Beccaria, 20
TORINO C.so Grosseto, 209
TORINO Via Chivasso, 11
TORINO Via Tripoli, 179
TRENTO Via Sighele, 7/1
TREVIGLIO Via G Mazzini, 10/B
TRIESTE via F Saverio, 138
UDINE Via Tavagnacco, 89/91
VARESE Via Carrobbio, 13
VERONA Via Pontiere, 2
VIAREGGIO Via A Volta,
VOGHERA P.zza G Carducci, 11

Desidero ricevere una copia omaggio del NUOVISSIMO CATALOGO ILLUSTRATO **REBIT** di ben 32 pagine: la più ampia e completa rassegna di computer, periferiche e accessori. Allego L. 2.000 per contributo spese di spedizione

Nome

Cognome

Via

Città C.A.P.

Data

Firma

SPEDIRE A: REBIT COMPUTER
CASELLA POSTALE 10488 - 20100 MILANO

Nella riga 10000 la variabile Z viene predisposta a 1 (codice errato). Nella riga 10010 si controlla la lunghezza del codice e nelle righe 10020-10100 si controlla il tipo (alfabetico o numerico) di ogni carattere del codice. Il ciclo FOR-NEXT delle linee 10150-10270 somma i valori corrispondenti ai caratteri di posto dispari e pari usando diverse funzioni di stringa, e infine le istruzioni 10280-10290 calcolano il resto delle divisione per 26, e lo convertono nel carattere di controllo. Se il codice è corretto, la variabile Z viene posta a 0.

Conclusioni

In un certo senso, si può dire che la routine presentata controlla l'esattezza "sintattica" del codice fiscale, cioè verifica che sia stato trascritto senza errori, nei limiti in cui il carattere di controllo consente tale verifica. Altro problema è quello di controllare l'esattezza "semantica" del codice, verificare cioè se corrisponde ai dati anagrafici del soggetto (che quindi devono essere dati in input al programma). Quest'ultimo è in effetti l'algoritmo di assegnazione del codice fiscale, che non ha interesse pratico per l'utente.

Alcune considerazioni sulle istruzioni del programma.

Alcuni interpreti richiedono il dimensionamento delle stringhe. In tal caso bisogna dichiarare DIM A$16.

La fuzione di stringa MID$(A$,I,1) restituisce l'I-esimo carattere della stringa A$, ed è presente nella gran parte degli interpreti Basic (per esempio Applesoft, PET Basic, TRS-80 Level II). In Integer Basic e con altri interpreti bisogna tradurre con A$(I,I). Nel Basic del Sinclair ZX80 si deve scrivere A$ (I TO I).

VAL(B$) converte stringhe di caratteri numerici in variabili numeriche. Per esempio VAL("12") è 12.

ASC(B$) converte il primo carattere di B$ nel suo codice ASCII. Generalmente i codici ASCII per le lettere A-Z partono da 65.

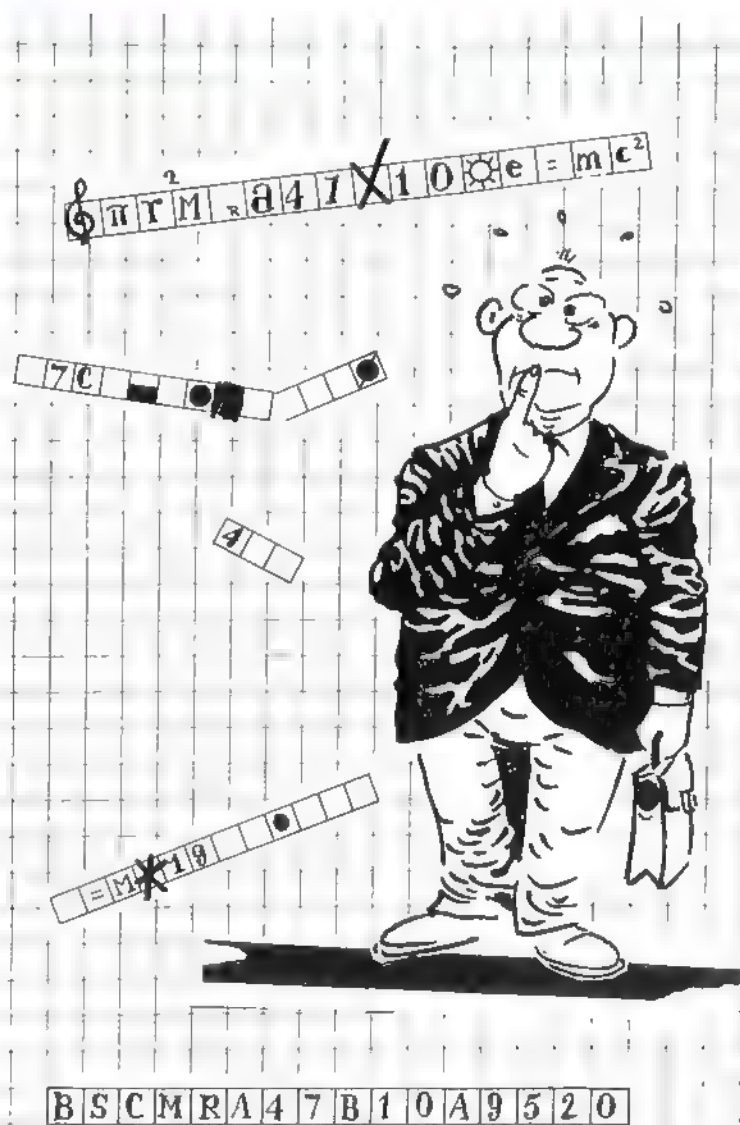
CHR$(R) è la funzione inversa di ASC(B$) e restituisce il carattere il cui codice è R.

Se il codice ASCII del vostro computer non codifica le lettere a partire dal numero 65, modificate di conseguenza le linee 10200, 10260, 10290. ■

Invitiamo tutti coloro che implementeranno questa routine sui loro computer, ad inviare eventuali correzioni, modifiche o suggerimenti a

Personal Software
Rubrica "Raccolta di routine Basic"
Via Rosellini, 12
20124 MILANO

I suggerimenti ricevuti saranno raccolti e pubblicati nei prossimi numeri della rubrica.

Nel prossimo numero:
il controllo della partita IVA.

Package di aritmetica intera per il 6502

di *Matteo Cerofolini*

La prima parte di questo articolo presenta un insieme di routine che implementano l'aritmetica intera in tutti i computer che usano il microprocessore 6502 (Apple II, Apple III, Pet, Vic, Commodore 64, AIM 65, Atom, BBC, SYM, Amico ecc.). Si può senz'altro affermare che la maggior parte dei personal e home computer diffusi in Italia (con l'unica eccezione per lo ZX80 e ZX81) usino il 6502. Le capacità di questo microprocessore sono molto elevate e, per la versione ad 1 MHz, possono essere paragonate alle capacità di uno Z80 a 4 MHz (vedi i test nella seconda parte dell'articolo). Inoltre è dotato di un set di istruzioni semplice ma molto potente e di un grande numero di modi di indirizzamento. Queste caratteristiche sono molto apprezzate dai progettisti dei nuovi microprocessori e sono alla base del successo del nuovo micro della Motorola: il 68000.

La seconda parte dell'articolo si rivolge in particolare ai possessori dell'Apple II. Viene presentata una applicazione delle routine di aritmetica intera descritte nella prima parte dell'articolo e viene fatto un confronto nella velocità di esecuzione e nella occupazione di memoria della stessa applicazione scritta nei vari linguaggi disponibili per l'Apple II.

Le routine qui presentate sono state studiate e messe a punto in occasione della realizzazione del minicompilatore Basic pubblicato su *Bit* (settembre 1981). In quell'occasione esse furono presentate prive di commenti poiché dovevano servire solo da supporto al compilatore e ne costituivano le routine di *run time*. Nonostante la struttura generale sia rimasta invariata esse sono state modificate in alcuni punti per renderle più generali e soprattutto sono stati inseriti numerosi commenti per facilitarne la comprensione.

La rappresentazione interna dei numeri è in *complemento a due* e per ogni numero vengono usati due byte di memoria. Il byte più significativo è nella locazione di memoria più bassa con il bit più significativo che costituisce il segno. Questa rappresentazione è una semplice estensione della rappresentazione ad 8 bit usata per le normali operazioni sull'accumulatore. Sfortunatamente ciò porta ad un piccolo inconveniente. Il più piccolo numero rappresentabile è −32768 mentre il più grande è +32767. Quindi se viene negato il valore −32768 ne nasce una condizione di overflow.

Le routine principali sono quattro: moltiplicazione intera, divisione intera, conversione da stringa a numero intero e conversione da numero intero a stringa. Queste quattro routine usano a loro volta altre subroutine che sono state inserite tra le routine principali. I punti di ingresso delle quattro routine principali sono i seguenti:

BINDEC	conversione da numero intero a stringa
DECBIN	conversione da stringa a numero intero
MOLT	moltiplicazione intera
DIV	divisione intera

Tutte le routine principali escono con il flag di *carry* azzerato se non vi è stato overflow durante l'operazione e con il flag attivato se vi è stata una condizione di overflow.

Tutto il package presenta condizioni di utilizzazione uniformi

Da notare che le routine DIV e MOLT richiedono gli operandi nella forma inversa a quella normalmente usata. Infatti, mentre di solito il numero intero è nella forma *low-high* cioè parte meno significativa nella locazione bassa di memoria e parte più significativa nella locazione alta di memoria, gli operandi PROD, MPCD, DVND e DVSR vanno caricati in maniera opposta. Per evitare confusione, si è preferito inserire all'ingresso del-

le due routine lo scambio dei byte così da avere delle condizioni di utilizzazione uniformi in tutto il package.

Le prestazioni

Per verificare le prestazioni del package descritto, è stato effettuato un test usando alcuni dei linguaggi disponibili sull'Apple II, e precisamente:

- Assembler 6502 che usa direttamente le routine presentate;
- Applesoft interpretato;
- Integer Basic interpretato;
- Applesoft compilato con il mini-compilatore pubblicato su *Bit*;
- Applesoft compilato con i compilatori più diffusi (Tasc, Hayden, Expediter, Speedstar).
- MBasic usato con il sistema operativo CP/M e la scheda Z80 della Microsoft.

È stata inoltre usata una scheda col microprocessore 8088 (ALF AD 8088 Processor Card) a 16 bit, che intercetta le routine matematiche dell'Applesoft per eseguirle molto più rapidamente. È una scheda molto interessante che permette anche l'uso del CP/M 86 e del MS-DOS sull'Apple II.

Le prestazioni del package sono state verificate con dodici linguaggi diversi

A titolo di curiosità si è provato il programma di test anche su un elaboratore di grandi dimensioni per verificare la differenza di prestazioni esistente tra un micropro-

Listato 1. *L'assemblato del package di aritmetica intera per il 6502 (segue). La parte finale del listato contiene il programma di test (dalla riga 7610).*

```
                  1010 *------------------------------
                  1020 * LE ROUTINES CHE SEGUONO ESTEN-
                  1030 * DONO LE CAPACITA' ARITMETICHE
                  1040 * DEL MICROPROCESSOR 6502 E
                  1050 * PERMETTONO L'USO DI ARITMETICA
                  1060 * INTERA SU QUANTITA' A 16 BIT
                  1070 * SEGNATE. L'ARITMETICA USATA E'
                  1080 * QUELLA DEL COMPLEMENTO A DUE.
                  1090 * IL RANGE DEI VALORI AMMESSI E'
                  1100 *
                  1110 *   -32767       +32767
                  1120 *
                  1130 * PER USARE LE ROUTINES CARICARE
                  1140 * I REGISTRI DI PAGINA ZERO CON
                  1150 * GLI OPPORTUNI OPERANDI E CHIA-
                  1160 * MARE LA ROUTINE OPPORTUNA. IL
                  1170 * RISULTATO VIENE FORNITO NEL
                  1180 * REGISTRO INDICATO. IN CASO DI
                  1190 * OVERFLOW IL CARRY AL RIENTRO E'
                  1200 * DIVERSO DA ZERO. LA DECISIONE
                  1210 * DA PRENDERE IN CASO DI OVERFLOW
                  1220 * E' QUINDI LASCIATA ALL'UTENTE
                  1230 * CHE PROBABILMENTE STAMPERA' UN
                  1240 * MESSAGGIO E RITORNERA' AL
                  1250 * PROPRIO MONITOR. OLTRE ALLE
                  1260 * ROUTINES NUMERICHE, VENGONO
                  1270 * FORNITE ANCHE DUE ROUTINES DI
                  1280 * UTILITA' PER LA CONVERSIONE
                  1290 * STRINGA-NUMERO E NUMERO-STRINGA
                  1300 * TUTTE LE ROUTINES RITORNANO CON
                  1310 * CARRY CLEAR SE NON VI SONO
                  1320 * STATI ERRORI ALTRIMENTI VIENE
                  1330 * SETTATO IL CARRY.
                  1340 *------------------------------
                  1350 * LE ROUTINE DERIVANO DA UN
                  1360 * INSIEME DI ROUTINE GIA' PRESEN-
                  1370 * TATE IN OCCASIONE DELLA PUBBLI-
                  1380 * CAZIONE DEL MINI-COMPILATORE
                  1390 * BASIC E SONO STATE MIGLIORATE
                  1400 * E RITOCCATE IN ALCUNI PUNTI.
                  1410 * NONOSTANTE SI SIA USATO UN
                  1420 * APPLE II PER LO SVILUPPO DELLE
                  1430 * ROUTINES ESSE SONO USABILI SU
                  1440 * QUALSIASI COMPUTER CHE USI IL
                  1450 * MICROPROCESSOR 6502.
                  1460 *
                  1470 *
                  1480 *------------------------------
                  1490        .OR $800
                  1500        .TF B.INTEGER
0800- 4C 5B 0A    1510        JMP INIZIO
                  1520 *------------------------------
                  1530 *
                  1540 *
                  1550 *------------------------------
                  1560 *
                  1570 * EQUATES IN PAGINA ZERO
                  1580 *
                  1590 *------------------------------
00EB-             1600 N      .EQ $EB
00ED-             1610 REG1   .EQ $ED
0006-             1620 N1     .EQ $06
0008-             1630 N2     .EQ $08
                  1640 *------------------------------
                  1650 *
                  1660 * REDEFINES PAGINA ZERO
                  1670 *
                  1680 *------------------------------
```

cessore e un maxi-elaboratore. Il computer è un Honeywell DPS serie 80 ed il linguaggio usato è il Basic presente nel sottosistema Time Sharing (TSS).

I programmi usati per il test sono riportati nei listati da 2 a 5 e al termine del listato 1 (dalla riga 7610). Lo Speed-Asm è uno speciale linguaggio di tipo Basic che permette, in unione all'assemblatore Lisa, una velocità quasi uguale a quella del linguaggio macchina. La versione usata per la prova con l'MBasic e col computer Honeywell DPS è uguale a quella Applesoft. I programmi sono sostanzialmente uguali in tutte le versioni di Basic ad eccezione di quella relativa al minicompilatore, in cui non è implementata l'istruzione FOR-NEXT che è dunque stata sostituita con una costruzione basata su IF-GOTO.

Da notare infine che nei due compilatori Microsoft e Hayden non si è usata l'opzione di compilazione con numeri interi che avrebbe fornito un programma-oggetto molto più veloce in termini di tempi di esecuzione.

I programmi di test sono sostanzialmente uguali in tutte le versioni di Basic

La tabella 1 riporta i tempi di esecuzione e l'occupazione di memoria di ciascuna versione di programma. Per quanto riguarda l'occupazione di memoria essa è data in settori di dischetto (256 byte) e non tiene conto dell'occupazione di memoria delle variabili nel caso di linguaggi interpretati e della necessità della presenza dell'interprete per molti dei linguaggi presentati.

Conclusioni

I risultati delle prove effettuate dimostrano che se è necessaria una

Segue listato 1.

```
00F9-             1690 PROD    .EQ $F9
00FD-             1700 MPCD    .EQ $FD
00F9-             1710 DVND    .EQ $F9
00FD-             1720 DVSR    .EQ $FD
                  1730 *
                  1740 *
                  1750 *
                  1760 *-------------------------------
                  1770 *
                  1780 *        ==== DECBIN ====
                  1790 *
                  1800 * CONVERSIONE STRINGA -> NUMERO
                  1810 *
                  1820 * PARAMETRI:
                  1830 *
                  1840 * REG1 - PUNTA ALLA STRINGA DA
                  1850 *        CONVERTIRE. I VALORI
                  1860 *        AMMESSI NELLA STRINGA
                  1870 *        SONO I NUMERI DA 0 A 9
                  1880 *        ED I SEGNI "+" E "-". IL
                  1890 *        SEGNO "+" E' OPZIONALE.
                  1900 *        IL PRIMO CARATTERE DI-
                  1910 *        VERSO DA '0' O '9' DE-
                  1920 *        TERMINA LA FINE DELLA
                  1930 *        STRINGA DA CONVERTIRE.
                  1940 *        ESSO PUO' ESSERE AD E-
                  1950 *        SEMPIO UNO ZERO BINARIO
                  1960 *        O UN CARRIAGE RETURN.
                  1970 *
                  1980 * N      - RISULTATO DELLA
                  1990 *          CONVERSIONE
                  2000 *-------------------------------
                  2010 DECBIN
                  2020 *
                  2030 * AZZERA IL CAMPO N
                  2040 *
0803- A9 00       2050        LDA #0
0805- 85 EB       2060        STA N
0807- 85 EC       2070        STA N+1
                  2080 *
                  2090 * SE LA STRINGA E' PRECEDUTA DA
                  2100 * UN SEGNO '-' O '+' SALTA IL
                  2110 * CARATTERE
                  2120 *
0809- A0 00       2130        LDY #$00
080B- B1 ED       2140        LDA (REG1),Y
080D- C9 AD       2150        CMP #'-+$80   '-' ?
080F- F0 04       2160        BEQ .1
0811- C9 AB       2170        CMP #'++$80   '+' ?
0813- D0 01       2180        BNE CNV
0815- C8          2190 .1     INY
                  2200 CNV
0816- B1 ED       2210        LDA (REG1),Y
0818- 20 37 08    2220        JSR DCMLCK    DECIMALE ?
081B- B0 0D       2230        BCS EOCNV     FINE CONVERS.
081D- 8D EA 09    2240        STA NEW
0820- 20 45 08    2250        JSR DBTEN
0823- B0 04       2260        BCS .2        OVERFLOW!!
0825- C8          2270        INY
0826- D0 EE       2280        BNE CNV
0828- 38          2290        SEC           ERRORE!!
                  2300 .2
0829- 60          2310        RTS
                  2320 EOCNV
                  2330 *
                  2340 * FINE DELLA CONVERSIONE. SE IL
                  2350 * PRIMO CARATTERE DELLA STRINGA
                  2360 * ERA IL SEGNO '-' SI ESEGUE IL
```

Linguaggio	Tempo di esecuzione (sec)	Con AD8088 (sec)	Occupazione di disco (settori)
Honeywell DPS	0.3	—	—
Assembler 6502	11.1	—	2
Minicompilatore	11.2	—	5
Speed ASM + Lisa	12.5	—	
Apple Pascal	15.2	—	
Compilatore Hayden	28.0	16.2	6
Compilatore Expediter	29.5	17.1	3
Compilatore Tasc	29.6	17.8	3+17
Compilatore Speedstar	31.9	20.6	11
Integer Basic	45.1	—	2
Applesoft	54.6	41.8	2
MBasic (Z80)	98.0	—	4

Tabella 1.

```
                 2370 * COMPLEMENTO A DUE DEL RISULTATO
                 2380 *
082A- A0 00      2390        LDY #0
082C- B1 ED      2400        LDA (REG1),Y
082E- C9 AD      2410        CMP #'-+$80    '-' ?
0830- D0 03      2420        BNE FINEC
0832- 20 76 08   2430        JSR TWOCOMP
                 2440 FINEC
0835- 18         2450        CLC           NO ERRORE
0836- 60         2460        RTS
                 2470 *
                 2480 *
                 2490 * QUESTA ROUTINE CONTROLLA CHE IL
                 2500 * CONTENUTO DELL'ACCUMULATORE SIA
                 2510 * NUMERICO. SE CIO' NON SI VERI-
                 2520 * FICA VIENE SETTATO IL FLAG DI
                 2530 * DI CARRY. SE SI TRATTA DI UNA
                 2540 * CIFRA VIENE RESTITUITO SOLO
                 2550 * IL SEMIBYTE MENO SIGNIFICATIVO
                 2560 *
0837- C9 B0      2570 DCMLCK CMP #$B0      < 0 ?
0839- 90 08      2580        BCC NDEC
083B- C9 BA      2590        CMP #$BA      > 9 ?
083D- B0 04      2600        BCS NDEC
083F- 29 0F      2610        AND #$F       SOLO NIBBLE
0841- 18         2620        CLC
0842- 60         2630        RTS
0843- 38         2640 NDEC   SEC
0844- 60         2650        RTS
                 2660 *
                 2670 * QUESTA ROUTINE MOLTIPLICA X 10
                 2680 * IL VALORE IN N E N+1 E SOMMA
                 2690 * AD N N+1 IL CONTENUTO DI NEW
                 2700 * SE VI E' OVERFLOW LA ROUTINE
                 2710 * RITORNA COL CARRY SET
                 2720 *
0845- 06 EB      2730 DBTEN  ASL N
0847- 26 EC      2740        ROL N+1
0849- B0 2A      2750        BCS EOMP      OVERFLOW
084B- A5 EB      2760        LDA N
084D- A6 EC      2770        LDX N+1
084F- 06 EB      2780        ASL N
0851- 26 EC      2790        ROL N+1
0853- B0 20      2800        BCS EOMP      OVERFLOW
0855- 06 EB      2810        ASL N
0857- 26 EC      2820        ROL N+1
0859- B0 1A      2830        BCS EOMP      OVERFLOW
```

grande velocità di esecuzione unita ad una bassa occupazione di memoria occorre usare il linguaggio macchina. Anche il minicompilatore si difende egregiamente e batte, in velocità di esecuzione e occupazione di memoria, i più blasonati compilatori Microsoft e Hayden. Occorre comunque ricordare che questi ultimi sono dei compilatori in grado di compilare la quasi totalità delle istruzioni Applesoft, mentre il minicompilatore riesce a compilarne solo una piccola parte.

Se è necessaria la massima velocità di esecuzione occorre usare il linguaggio macchina

Un discorso a parte va fatto per i risultati ottenuti dal maxicomputer Honeywell DPS. Come si vede la capacità di calcolo di un grande elaboratore è di molto superiore a quella del pur veloce 6502. I tempi inoltre, sono stati presi mentre l'elaboratore era impegnato ad eseguire molte altre task e potevano essere molto migliori se il test fosse stato eseguito a macchina del tutto scarica. Per il calcolo di un tempo così breve si è eseguito un loop di 1000000 di FOR-NEXT e si è poi rapportato alle misure effettuate per gli altri linguaggi.

Vorrei, per concludere, precisare che i test di questo tipo hanno una validità relativa perché confrontano solo la velocità di esecuzione e l'occupazione di memoria e non tengono conto di altri fattori quali ad esempio la facilità di programmazione e di debugging, la leggibilità dei programmi, la completezza del set di istruzioni di un determinato linguaggio e così via.

Segue listato 1.

```
085B- 18        2840         CLC
085C- 65 EB     2850         ADC N
085E- 85 EB     2860         STA N
0860- 8A        2870         TXA
0861- 65 EC     2880         ADC N+1
0863- B0 10     2890         BCS EOMP     OVERFLOW
0865- AA        2900         TAX
0866- 85 EC     2910         STA N+1
0868- A5 EB     2920         LDA N
086A- 18        2930         CLC
086B- 6D EA 09  2940         ADC NEW
086E- 85 EB     2950         STA N
0870- 8A        2960         TXA
0871- 69 00     2970         ADC #0
0873- 85 EC     2980         STA N+1
                2990 EOMP
0875- 60        3000         RTS
                3010 *
                3020 * QUESTA ROUTINE ESEGUE IL
                3030 * COMPLEMENTO A 2 DEL DEL NUMERO
                3040 * CONTENUTO NEL CAMPO N E N+1
                3050 *
                3060 TWOCOMP
0876- A5 EB     3070         LDA N
0878- 49 FF     3080         EOR #$FF
087A- 85 EB     3090         STA N
087C- A5 EC     3100         LDA N+1
087E- 49 FF     3110         EOR #$FF
0880- 85 EC     3120         STA N+1
0882- E6 EB     3130         INC N
0884- D0 02     3140         BNE EXCOMP
0886- E6 EC     3150         INC N+1
                3160 EXCOMP
0888- 60        3170         RTS
                3180 *-------------------------------
                3190 *
                3200 *        ==== BINDEC ====
                3210 *
                3220 * CONVERSIONE NUMERO -> STRINGA
                3230 *
                3240 * PARAMETRI:
                3250 *
                3260 * N    - NUMERO BINARIO INTERO
                3270 *        DA CONVERTIRE
                3280 *
                3290 * REG1 - PUNTATORE DI PAGINA
                3300 *        ZERO CHE PUNTA ALLA
                3310 *        STRINGA ASCII RISULTATO
                3320 *        DELLA CONVERSIONE. LA
                3330 *        STRINGA TERMINA CON IL
                3340 *        VALORE BINARIO $00. IL
                3350 *        SEGNO '+' VIENE OMESSO
                3360 *        MENTRE IL SEGNO '-' VIE-
```

Segue listato 1.

Personalmente preferisco usare l'Applesoft interpretato integrandolo, ove occorra particolare velocità, con delle CALL a subroutine in linguaggio macchina. Ciò permette di usare la facilità di debugging del Basic e di sfruttare al massimo la velocità dell'unità centrale 6502.

Desidero infine ringraziare il dott. Rabellino di Alba per l'aiuto ed i preziosi consigli nell'effettuare alcuni test e la ditta Bits and Bytes di Milano per averci fornito i vari compilatori, lo Speed ASM, il Lisa e la scheda ALF "FTR 8088". ■

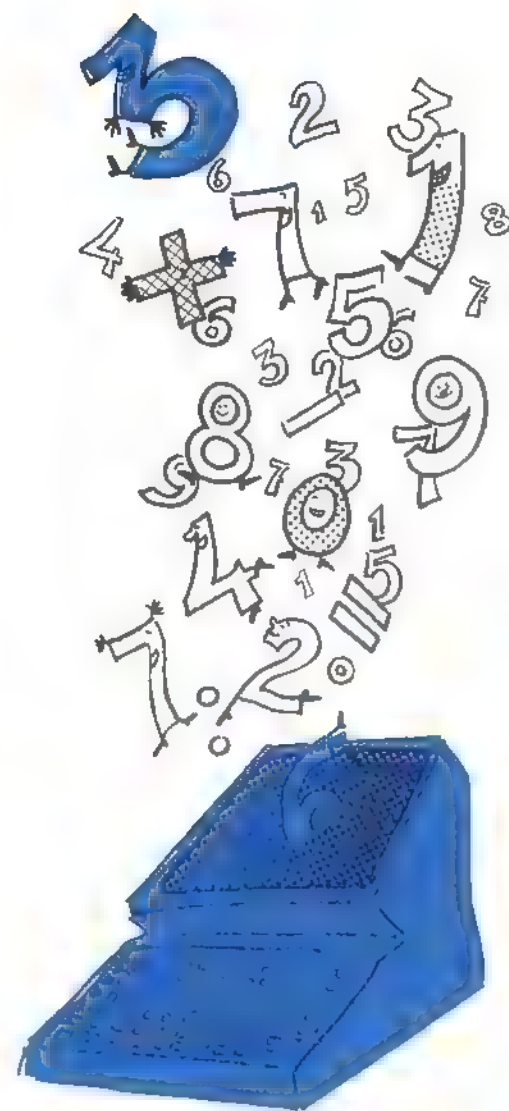
Bibliografia

MATTEO CEROFOLINI "Mini-Compilatore Basic per Apple II" *Bit* 20, settembre 1981.

BRUCE D. CARBREY "An Integer Math Package for the 8080" *Byte* 6, 5, may 1981.

W.J. WELLER *Practical Microcomputer Programming: The 6502* Norther Technology Book, 1980.

RODNAY ZAKS *Programmazione del 6502* Jackson, 1981.

ROBERT FINDLAY *6502 Software Gourmet Guide and Cookboook* Scelbi Publications, 1979.

Segue listato 1.

```
                  3370 *         INDICATO.
                  3380 *-------------------------------
                  3390 *
                  3400 BINDEC
                  3410 *
                  3420 * SALVA IL REGISTRO REG1
                  3430 *
0889- A5 ED       3440         LDA REG1
088B- 8D E8 09    3450         STA DEP
088E- A5 EE       3460         LDA REG1+1
0890- 8D E9 09    3470         STA DEP+1
                  3480 *
                  3490 * CONTROLLO SE NEGATIVO
                  3500 *
0893- A5 EC       3510         LDA N+1
0895- 10 0C       3520         BPL .1
                  3530 *
                  3540 * IL NUMERO IN N N+1 E' NEGATIVO
                  3550 * E QUINDI SI ESEGUE IL COMPLE-
                  3560 * MENTO A DUE E SI METTE IN
                  3570 * OUTPUT IL SEGNO '-'
                  3580 *
0897- A9 2D       3590         LDA #'-'
0899- A0 00       3600         LDY #0
089B- 91 ED       3610         STA (REG1),Y
089D- 20 E8 08    3620         JSR INCREG1  INCR. REG1
08A0- 20 76 08    3630         JSR TWOCOMP
                  3640 .1
08A3- A9 05       3650         LDA #5       MAX CIFRE
08A5- 8D EB 09    3660         STA CNT
                  3670 *
                  3680 * INDICE PER L'ACCESSO ALLA
                  3690 * TABELLA DI POTENZE DEL 10
                  3700 *
08A8- A0 00       3710         LDY #0
                  3720 DC1
                  3730 *
                  3740 * INIZIALIZZA DIGIT A ZERO ASCII
                  3750 *
08AA- A9 30       3760         LDA #'0'
08AC- 8D EC 09    3770         STA DIGIT
                  3780 DC2
                  3790 *
                  3800 * SOTTRAE LA POTENZA DEL 10
                  3810 * FINO AD AVERE ZERO O MENO E
                  3820 * INCREMENTA LA CIFRA DIGIT
                  3830 *
08AF- A5 EB       3840         LDA N
08B1- 38          3850         SEC
08B2- F9 F3 09    3860         SBC PT,Y
08B5- AA          3870         TAX
08B6- A5 EC       3880         LDA N+1
08B8- F9 F4 09    3890         SBC PT+1,Y
08BB- 90 09       3900         BCC DC3
08BD- 85 EC       3910         STA N+1
08BF- 86 EB       3920         STX N
08C1- EE EC 09    3930         INC DIGIT
08C4- D0 E9       3940         BNE DC2
                  3950 DC3
                  3960 *
                  3970 * METTE DIGIT IN MEMORIA PUNTATA
                  3980 * DA REG1 E INCREMENTA IL
                  3990 * IL REGISTRO REG1
                  4000 *
08C6- A2 00       4010         LDX #0
08C8- AD EC 09    4020         LDA DIGIT
08CB- 81 ED       4030         STA (REG1,X)
08CD- 20 E8 08    4040         JSR INCREG1
                  4050 DC4
                  4060 *
                  4070 * PUNTA ALLA POTENZA DEL 10
                  4080 * CHE SEGUE NELLA TABELLA PT
                  4090 *
08D0- C8          4100         INY
08D1- C8          4110         INY
                  4120 *
                  4130 * DECREMENTA IL CONTATORE DI
                  4140 * CIFRE E SE <> 0 RIPETE
                  4150 *
08D2- CE EB 09    4160         DEC CNT
08D5- D0 D3       4170         BNE DC1
                  4180 *
                  4190 * METTE $00 A FINE STRINGA
                  4200 *
08D7- A9 00       4210         LDA #0
08D9- AA          4220         TAX
08DA- 81 ED       4230         STA (REG1,X)
                  4240 *
                  4250 * RIMETTE IL REGISTRO REG1 AL
                  4260 * VALORE ORIGINARIO
                  4270 *
08DC- AD E8 09    4280         LDA DEP
08DF- 85 ED       4290         STA REG1
08E1- AD E9 09    4300         LDA DEP+1
08E4- 85 EE       4310         STA REG1+1
08E6- 18          4320         CLC          NO ERRORE
08E7- 60          4330         RTS
                  4340 *
                  4350 * INCREMENTA IL PUNTATORE DI
                  4360 * PAGINA ZERO REG1
                  4370 *
                  4380 INCREG1
08E8- E6 ED       4390         INC REG1
08EA- D0 02       4400         BNE .1
08EC- E6 EE       4410         INC REG1+1
                  4420 .1
08EE- 60          4430         RTS
                  4440 *-------------------------------
                  4450 *
                  4460 *       ==== MOLT =====
                  4470 *
                  4480 * MOLTIPLICAZIONE INTERA
                  4490 *
                  4500 * PARAMETRI:
                  4510 *
                  4520 * PROD - 4 BYTES PER IL MOLTIPLI-
                  4530 *        CATORE E IL PRODOTTO
                  4540 *
                  4550 * MPCD - 2 BYTES PER IL MOLTIPLI-
                  4560 *        CANDO
                  4570 *
                  4580 * INGRESSO:
                  4590 * MOLTIPLICATORE IN PROD+2 PROD+3
                  4600 * MOLTIPLICANDO IN  MPCD E MPCD+1
                  4610 * USCITA  :
                  4620 * PRODOTTO IN PROD FINO PROD+3
                  4630 *
                  4640 *-------------------------------
                  4650 MOLT
                  4660 *
                  4670 * ESEGUE LA NORMALIZZAZIONE DEGLI
                  4680 * OPERANDI E LA RICERCA DEL SEGNO
                  4690 *
                  4700 *
                  4710 * INVERTE GLI OPERANDI
                  4720 *
08EF  20 7D 09    4730         JSR SCAMBIO
                  4740 *
                  4750 * RENDE POSITIVI GLI OPERANDI
                  4760 *
08F2- 20 1D 09    4770         JSR NORMIN
                  4780 *
                  4790 * AZZERA LA PARTE ALTA DEL
                  4800 * RISULTATO
                  4810 *
08F5- A9 00       4820         LDA #0
08F7- 85 F9       4830         STA PROD
08F9  85 FA       4840         STA PROD+1
                  4850 UNSMO
                  4860 *
                  4870 * NUMERO MASSIMO DI BIT
                  4880 *
08FB- A2 11       4890         LDX #17
08FD- 18          4900         CLC
                  4910 UNSM1
08FE- 20 D4 09    4920         JSR SRQL
0901- CA          4930         DEX
0902- F0 12       4940         BEQ UNSM2    FINE 16 BITS
0904- 90 F8       4950         BCC UNSM1    BIT = 0
0906- A5 FA       4960         LDA PROD+1
0908- 18          4970         CLC
0909- 65 FE       4980         ADC MPCD+1
090B- 85 FA       4990         STA PROD+1
090D- A5 F9       5000         LDA PROD
090F- 65 FD       5010         ADC MPCD
0911- 85 F9       5020         STA PROD
0913- 4C FE 08    5030         JMP UNSM1    CONTINUA
                  5040 UNSM2
                  5050 *
                  5060 * ESEGUE LA NORMALIZZAZIONE DEL
                  5070 * RISULTATO IN USCITA
                  5080 *
0916- 20 40 09    5090         JSR NORMOT
0919- 20 7D 09    5100         JSR SCAMBIO
091C- 60          5110         RTS
                  5120 *
                  5130 * LA ROUTINE CHE SEGUE TROVA IL
                  5140 * SEGNO DEL RISULATO DELLA OPERA-
                  5150 * ZIONE E LO SALVA NEL CAMPO
                  5160 * SEGNO. GLI OPERANDI VENGONO POI
```

(segue)

Segue listato 1.

```
                5170 * TRASFORMATI IN INTERI POSITIVI
                5180 * FACENDONE IL COMPLEMENTO A DUE
                5190 *
                5200 NORMIN
                5210 *
                5220 * TROVA IL SEGNO DEL RISULTATO
                5230 *
091D- A5 FB     5240        LDA PROD+2
091F- 45 FD     5250        EOR MPCD
0921- 8D F2 09  5260        STA SEGNO
                5270 *
                5280 * ESEGUE IL COMPLEMENTO A 2 SE
                5290 * L'OPERANDO E' NEGATIVO
                5300 *
0924- A5 FB     5310        LDA PROD+2
0926- 10 10     5320        BPL SEGN2
0928- 49 FF     5330        EOR #$FF
092A- 85 FB     5340        STA PROD+2
092C- A5 FC     5350        LDA PROD+3
092E- 49 FF     5360        EOR #$FF
0930- 85 FC     5370        STA PROD+3
0932- E6 FC     5380        INC PROD+3
0934- D0 02     5390        BNE SEGN2
0936- E6 FB     5400        INC PROD+2
                5410 SEGN2
                5420 *
                5430 * ESEGUE IL COMPLEMENTO A DUE
                5440 * DELL'ALTRO OPERANDO SE QUESTO
                5450 * E' NEGATIVO
                5460 *
0938- A5 FD     5470        LDA MPCD
093A- 10 10     5480        BPL EXNORMIN
093C- 49 FF     5490        EOR #$FF
093E- 85 FD     5500        STA MPCD
0940- A5 FE     5510        LDA MPCD+1
0942- 49 FF     5520        EOR #$FF
0944- 85 FE     5530        STA MPCD+1
0946- E6 FE     5540        INC MPCD+1
0948- D0 02     5550        BNE EXNORMIN
094A- E6 FD     5560        INC MPCD
                5570 EXNORMIN
094C- 60        5580        RTS
                5590 *
                5600 *
                5610 * LA ROUTINE CHE SEGUE EFFETTUA
                5620 * LA NORMALIZZAZIONE DEL
                5630 * RISULTATO IN USCITA. CONTROLLA
                5640 * CHE NON VI SIA OVERFLOW E SE IL
                5650 * CAMPO SEGNO ERA NEGATIVO ESEGUE
                5660 * IL COMPLEMENTO A DUE DEL
                5670 * RISULTATO.
                5680 *
                5690 NORMOT
                5700 *
                5710 * CONTROLLA CHE NON VI SIA
                5720 * OVERFLOW
                5730 *
094D- A5 F9     5740        LDA PROD
094F- F0 03     5750        BEQ NORMOT1
0951- 4C 7B 09  5760        JMP OVERFLOW
                5770 NORMOT1
0954- A5 FA     5780        LDA PROD+1
0956- F0 03     5790        BEQ NORMOT2
0958- 4C 7B 09  5800        JMP OVERFLOW
                5810 NORMOT2
095B- A5 FB     5820        LDA PROD+2
095D- 10 03     5830        BPL NORMOT3
095F- 4C 7B 09  5840        JMP OVERFLOW
                5850 NORMOT3
                5860 *
                5870 * SE SEGNO NEGATIVO ESEGUE IL
                5880 * COMPLEMENTO A DUE DEL RISULTATO
                5890 *
0962- AD F2 09  5900        LDA SEGNO
0965- 10 12     5910        BPL EXNORMOT
0967- A5 FB     5920        LDA PROD+2
0969- 49 FF     5930        EOR #$FF
096B- 85 FB     5940        STA PROD+2
096D- A5 FC     5950        LDA PROD+3
096F- 49 FF     5960        EOR #$FF
0971- 85 FC     5970        STA PROD+3
0973- E6 FC     5980        INC PROD+3
0975- D0 02     5990        BNE EXNORMOT
0977- E6 FB     6000        INC PROD+2
                6010 EXNORMOT
0979- 18        6020        CLC
097A- 60        6030        RTS
                6040 *
                6050 * ROUTINE DI OVERFLOW
                6060 *
                6070 OVERFLOW
097B- 38        6080        SEC            SEGNALA OVER
097C- 60        6090        RTS
                6100 *
                6110 * SCAMBIA GLI OPERANDI DAL
                6120 * FORMATO LOW-HIGH A HIGH-LOW
                6130 * USANDO LO STACK COME DEPOSITO
                6140 *
                6150 SCAMBIO
097D- A5 FB     6160        LDA PROD+2
097F- 48        6170        PHA
0980- A5 FC     6180        LDA PROD+3
0982- 85 FB     6190        STA PROD+2
0984- 68        6200        PLA
0985- 85 FC     6210        STA PROD+3
0987- A5 FD     6220        LDA MPCD
0989- 48        6230        PHA
098A- A5 FE     6240        LDA MPCD+1
098C- 85 FD     6250        STA MPCD
098E- 68        6260        PLA
098F- 85 FE     6270        STA MPCD+1
0991- 60        6280        RTS
                6290 *----------------------------------
                6300 *
                6310 *       ==== DIV  =====
                6320 *
                6330 * DIVISIONE INTERA
                6340 *
                6350 * PARAMETRI:
                6360 *
                6370 * DVND - 4 BYTES PER IL DIVIDENDO
                6380 *         E IL QUOZIENTE
                6390 *
                6400 * DVSR - 2 BYTES PER IL DIVISORE
                6410 *
                6420 *
                6430 * INGRESSO:
                6440 * DIVIDENDO DVND+2 E DVND+3
                6450 * DIVISORE IN DVSR E DVSR+1
                6460 * USCITA  :
                6470 * QUOZIENTE IN DVND+2 E DVND+3
                6480 * RESTO IN DVND E DVND+1
                6490 *
                6500 *----------------------------------
                6510 *
                6520 DIV
                6530 *
                6540 * SCAMBIA GLI OPERANDI
                6550 *
0992- 20 7D 09  6560        JSR SCAMBIO
                6570 *
                6580 * NORMALIZZA GLI OPERANDI
                6590 *
0995- 20 1D 09  6600        JSR NORMIN
                6610 *
                6620 * AZZERA LA PARTE ALTA DEL
                6630 * RISULTATO
                6640 *
0998- A9 00     6650        LDA #0
099A- 85 F9     6660        STA DVND
099C- 85 FA     6670        STA DVND+1
                6680 *
                6690 * NUMERO MAX DI BITS = 17
                6700 *
099E- A2 11     6710        LDX #17
09A0- 18        6720        CLC
                6730 UNSDV1
09A1- A5 FA     6740        LDA DVND+1
09A3- 38        6750        SEC
09A4- E5 FE     6760        SBC DVSR+1
09A6- A8        6770        TAY
09A7- A5 F9     6780        LDA DVND
09A9- E5 FD     6790        SBC DVSR
09AB- 90 05     6800        BCC UNSDV2
09AD- 85 F9     6810        STA DVND
09AF- 98        6820        TYA
09B0- 85 FA     6830        STA DVND+1
                6840 UNSDV2
09B2- 20 DD 09  6850        JSR RLQL
09B5- CA        6860        DEX
09B6- D0 E9     6870        BNE UNSDV1
                6880 *
                6890 * NORMALIZZA IL SEGNO IN USCITA
                6900 * ED ESEGUE, SE NEGATIVO IL
                6910 * COMPLEMENTO A DUE DEL RISULTATO
                6920 *
09B8- AD F2 09  6930        LDA SEGNO
09BB- 10 12     6940        BPL EXDIV
```

(segue)

Segue listato 1.

```
09BD- A5 FB     6950        LDA PROD+2
09BF- 49 FF     6960        EOR #$FF
09C1- 85 FB     6970        STA PROD+2
09C3- A5 FC     6980        LDA PROD+3
09C5- 49 FF     6990        EOR #$FF
09C7- 85 FC     7000        STA PROD+3
09C9- E6 FC     7010        INC PROD+3
09CB- D0 02     7020        BNE EXDIV
09CD- E6 FB     7030        INC PROD+2
                7040 EXDIV
09CF- 20 7D 09  7050        JSR SCAMBIO
09D2- 18        7060        CLC
09D3- 60        7070        RTS
                7080 *
                7090 * SUBROUTINE DI SHIFT QUADRUPLO
                7100 * A DESTRA. ESCE CON IL BIT
                7110 * SHIFTATO NEL CARRY.
                7120 *
09D4- 66 F9     7130 SRQL   ROR PROD
09D6- 66 FA     7140        ROR PROD+1
09D8- 66 FB     7150        ROR PROD+2
09DA- 66 FC     7160        ROR PROD+3
09DC- 60        7170        RTS
                7180 *
                7190 * SUBROUTINE DI SHIFT QUADRUPLO
                7200 * A SINISTRA. ESCE CON IL BIT
                7210 * SHIFTATO NEL CARRY.
                7220 *
09DD- 26 FC     7230 RLQL   ROL PROD+3
09DF- 26 FB     7240        ROL PROD+2
09E1- 26 FA     7250        ROL PROD+1
09E3- 26 F9     7260        ROL PROD
09E5- 60        7270        RTS
                7280 *
                7290 * AREE DI LAVORO.QUESTE AREE DI
                7300 * MEMORIA POSSONO ESSERE MESSE IN
                7310 * QUALUNQUE LOCAZIONE. E' BENE
                7320 * NOTARE COMUNQUE CHE, SE MESSE
                7330 * IN PAGINA ZERO, RENDONO PIU'
                7340 * COMPATTE E PIU' VELOCI LE
                7350 * ROUTINES PRESENTATE.
                7360 *
09E6-           7370 OVER   .BS 1
09E7-           7380 ERROR  .BS 1
09E8-           7390 DEP    .BS 2
09EA-           7400 NEW    .BS 1
09EB-           7410 CNT    .BS 1
09EC-           7420 DIGIT  .BS 1
09ED-           7430 DECNUM .BS 5
09F2-           7440 SEGNO  .BS 1
                7450 *
                7460 * TABELLA DI POTENZE DEL 10.
                7470 * VIENE USATA DALLA ROUTINE DI
                7480 * CONVERSIONE BINARIO-DECIMALE
                7490 *
09F3- 10 27     7500 PT     .DA 10000
09F5- E8 03     7510        .DA 1000
09F7- 64 00     7520        .DA 100
09F9- 0A 00     7530        .DA 10
09FB- 01 00     7540        .DA 1
                7550 *
                7560 *
                7570 *
                7580 *-------------------------------
                7590 *-------------------------------
                7600 *-------------------------------
                7610 * INIZIO PROGRAMMA DI TEST
                7620 *-------------------------------
                7630 *-------------------------------
                7640 *-------------------------------
                7650 *
                7660 *
                7670 *
                7680 *-------------------------------
                7690 * EQUATES
                7700 *-------------------------------
                7710 *
DB3A-           7720 PRTSTR .EQ $DB3A    STAMPA STRING
0200-           7730 KEYBRD .EQ $200     TASTIERA
FD6A-           7740 GETLN  .EQ $FD6A    GET LINE
FC58-           7750 HOME   .EQ $FC58    HOME
FD8E-           7760 CRLF   .EQ $FD8E    RETURN
                7770 *
                7780 *-------------------------------
                7790 * LITERALS
                7800 *-------------------------------
                7810 *
09FD- D1 D5 C1
0A00- CE D4 C5
0A03- A0 D6 CF
0A06- CC D4 C5
0A09- A0        7820 RICH0  .AS -"QUANTE VOLTE "
0A0A- 00        7830        .HS 00
0A0B- D0 D2 C9
0A0E- CD CF A0
0A11- CE D5 CD
0A14- C5 D2 CF
0A17- A0        7840 RICH1  .AS -"PRIMO NUMERO "
0A18- 00        7850        .HS 00
0A19- D3 C5 C3
0A1C- CF CE C4
0A1F- CF A0 CE
0A22- D5 CD C5
0A25- D2 CF A0  7860 RICH2  .AS -"SECONDO NUMERO "
0A28- 00        7870        .HS 00
0A29- CF D6 C5
0A2C- D2 C6 CC
0A2F- CF D7 A0
0A32- A1 A1 A1  7880 RISP1  .AS -"OVERFLOW !!!"
0A35- 00        7890        .HS 00
0A36- D2 C9 D3
0A39- D5 CC D4
0A3C- C1 D4 CF
0A3F- A0 BD A0  7900 RISP2  .AS -"RISULTATO = "
0A42- 00        7910        .HS 00
                7920 *
                7930 *-------------------------------
                7940 * CONTATORI
                7950 *-------------------------------
                7960 *
0A43-           7970 MAX    .BS 2
0A45-           7980 CTR    .BS 2
                7990 *
                8000 *-------------------------------
                8010 * DEPOSITI NUMERI
                8020 *-------------------------------
                8030 *
                8040 *N1    .BS 2
                8050 *N2    .BS 2
                8060 *
                8070 *-------------------------------
                8080 *
                8090 *
                8100 *
0A47-           8110 BUFF   .BS 20
                8120 *-------------------------------
                8130 * INIZIO TEST
                8140 *-------------------------------
                8150 *
                8160 * PULIZIA VIDEO
                8170 *-------------------------------
0A5B- 20 58 FC  8180 INIZIO JSR HOME
0A5E- A9 BF     8190        LDA #'?+$80
0A60- 85 33     8200        STA $33
                8210 INIZIO1
0A62- 20 8E FD  8220        JSR CRLF
                8230 *-------------------------------
                8240 * RICHIESTA NUMERO VOLTE
                8250 *-------------------------------
0A65- A9 FD     8260        LDA #RICH0
0A67- A0 09     8270        LDY /RICH0
0A69- 20 3A DB  8280        JSR PRTSTR
                8290 *-------------------------------
                8300 * LEGGE DA TASTIERA
                8310 *-------------------------------
0A6C- 20 6A FD  8320        JSR GETLN
0A6F- 20 8E FD  8330        JSR CRLF
                8340 *-------------------------------
                8350 * TRASFORMA IN INTERO IN MAX
                8360 *-------------------------------
0A72- A9 00     8370        LDA #KEYBRD
0A74- 85 ED     8380        STA REG1
0A76- A9 02     8390        LDA /KEYBRD
0A78- 85 EE     8400        STA REG1+1
0A7A- 20 03 0B  8410        JSR DECBIN
0A7D- 90 03     8420        BCC .1
0A7F- 4C 0F 0B  8430        JMP ERRORE    OVERFLOW ?
                8440 *-------------------------------
                8450 * SALVA IN MAX
                8460 *-------------------------------
0A82- A5 EB     8470 .1     LDA N
0A84- 8D 43 0A  8480        STA MAX
0A87- A5 EC     8490        LDA N+1
0A89- 8D 44 0A  8500        STA MAX+1
                8510 *-------------------------------
                8520 * RICHIESTA PRIMO NUMERO
                8530 *-------------------------------
0A8C- A9 0B     8540        LDA #RICH1
```

(segue)

Segue listato 1.

```
0A8E- A0 0A    8550        LDY /RICH1
0A90- 20 3A DB 8560        JSR PRTSTR
               8570 *--------------------------------
               8580 * LETTURA DA TASTIERA
               8590 *--------------------------------
0A93- 20 6A FD 8600        JSR GETLN
0A96- 20 8E FD 8610        JSR CRLF
               8620 *--------------------------------
               8630 * TRASFORMA IN INTERO IN N,N+1
               8640 *--------------------------------
0A99- A9 00    8650        LDA #KEYBRD
0A9B- 85 ED    8660        STA REG1
0A9D- A9 02    8670        LDA /KEYBRD
0A9F- 85 EE    8680        STA REG1+1
0AA1- 20 03 08 8690        JSR DECBIN
0AA4- B0 69    8700        BCS ERRORE   OVERFLOW ?
               8710 *--------------------------------
               8720 * SALVA IL NUMERO IN N1,N1+1
               8730 *--------------------------------
0AA6- A5 EB    8740        LDA N
0AA8- 85 06    8750        STA N1
0AAA- A5 EC    8760        LDA N+1
0AAC- 85 07    8770        STA N1+1
               8780 *--------------------------------
               8790 * RICHIESTA SECONDO NUMERO
               8800 *--------------------------------
0AAE- A9 19    8810        LDA #RICH2
0AB0- A0 0A    8820        LDY /RICH2
0AB2- 20 3A DB 8830        JSR PRTSTR
               8840 *--------------------------------
               8850 * LETTURA DA TASTIERA
               8860 *--------------------------------
0AB5- 20 6A FD 8870        JSR GETLN
0AB8- 20 8E FD 8880        JSR CRLF
               8890 *--------------------------------
               8900 * TRASFORMAZIONE IN NUMERO INTERO
               8910 *--------------------------------
0ABB- A9 00    8920        LDA #KEYBRD
0ABD- 85 ED    8930        STA REG1
0ABF- A9 02    8940        LDA /KEYBRD
0AC1- 85 EE    8950        STA REG1+1
0AC3- 20 03 08 8960        JSR DECBIN
0AC6- B0 47    8970        BCS ERRORE   OVERFLOW?
               8980 *--------------------------------
               8990 * SALVA IL RISULTATO IN N2,N2+1
               9000 *--------------------------------
0AC8- A5 EB    9010        LDA N
0ACA- 85 08    9020        STA N2
0ACC- A5 EC    9030        LDA N+1
0ACE- 85 09    9040        STA N2+1
               9050 *--------------------------------
               9060 * CARICA IL CONTATORE
               9070 *--------------------------------
0AD0- AD 43 0A 9080        LDA MAX
0AD3- 8D 45 0A 9090        STA CTR
0AD6- AD 44 0A 9100        LDA MAX+1
0AD9- 8D 46 0A 9110        STA CTR+1
               9120 *--------------------------------
               9130 * LOOP PRINCIPALE
               9140 * AZZERA CAMPI
               9150 * CARICA GLI OPERANDI
               9160 *--------------------------------
               9170 LOOP
0ADC- A5 06    9180        LDA N1
0ADE- 85 FB    9190        STA PROD+2
0AE0- A5 07    9200        LDA N1+1
0AE2- 85 FC    9210        STA PROD+3
0AE4- A5 08    9220        LDA N2
0AE6- 85 FD    9230        STA MPCD
0AE8- A5 09    9240        LDA N2+1
0AEA- 85 FE    9250        STA MPCD+1
               9260 *--------------------------------
               9270 * ESEGUE LA MOLTIPLICAZIONE
               9280 *--------------------------------
0AEC- 20 EF 08 9290        JSR MOLT
0AEF- B0 1E    9300        BCS ERRORE   OVERFLOW ?
               9310 *--------------------------------
               9320 * DECREMENTA IL CONTATORE
               9330 * E RIPETE PER MAX VOLTE
               9340 *--------------------------------
0AF1- CE 45 0A 9350        DEC CTR
0AF4- D0 E6    9360        BNE LOOP
0AF6- AD 46 0A 9370        LDA CTR+1
0AF9- F0 06    9380        BEQ STAMPA
0AFB- CE 46 0A 9390        DEC CTR+1
0AFE- 4C DC 0A 9400        JMP LOOP
               9410 *-----  --------------------------
               9420 * FINE TEST, STAMPA RISULTATO
               9430 *--------------------------------
               9440 STAMPA
0B01- A5 FB    9450        LDA PROD+2
0B03- 85 EB    9460        STA N
0B05- A5 FC    9470        LDA PROD+3
0B07- 85 EC    9480        STA N+1
0B09- 20 19 0B 9490        JSR PRINTN
0B0C- 4C 62 0A 9500        JMP INIZIO1
               9510 *--------------------------------
               9520 * ROUTINE DI ERRORE
               9530 *--------------------------------
               9540 ERRORE
0B0F- A9 29    9550        LDA #RISP1
0B11- A0 0A    9560        LDY /RISP1
0B13- 20 3A DB 9570        JSR PRTSTR
0B16- 4C 62 0A 9580        JMP INIZIO1
               9590 *--------------------------------
               9600 * STAMPA IL CONTENUTO DI N,N+1
               9610 *-----  -------------------------
               9620 PRINTN
0B19- A9 36    9630        LDA #RISP2
0B1B- A0 0A    9640        LDY /RISP2
0B1D- 20 3A DB 9650        JSR PRTSTR
0B20- A9 47    9660        LDA #BUFF
0B22- 85 ED    9670        STA REG1
0B24- A9 0A    9680        LDA /BUFF
0B26- 85 EE    9690        STA REG1+1
0B28- 20 89 08 9700        JSR BINDEC
0B2B- A5 ED    9710        LDA REG1
0B2D- A4 EE    9720        LDY REG1+1
0B2F- 20 3A DB 9730        JSR PRTSTR
0B32- 60       9740        RTS
```

Tabella dei simboli del listato 1.

```
0889- BINDEC
.01=08A3
0A47- BUFF
09EB- CNT
0816- CNV
.02=0829
FD8E- CRLF
0A45- CTR
0845- DBTEN
08AA- DC1
08AF- DC2
08C6- DC3
08D0- DC4
0837- DCMLCK
0803- DECBIN
.01=0815
09ED- DECNUM
09E8- DEP
09EC- DIGIT
0992- DIV
00F9- DVND
00FD- DVSR
082A- EOCNV
0875- EOMP
09E7- ERROR
0B0F- ERRORE
0888- EXCOMP
09CF- EXDIV
094C- EXNORMIN
0979- EXNORMOT
0835- FINEC
FD6A- GETLN
FC58- HOME
08E8- INCREG1
.01=08EE
0A5B- INIZIO
0A62- INIZIO1
.01=0AB2
0200- KEYBRD
0ADC- LOOP
0A43- MAX
08EF- MOLT
00FD- MPCD
00EB- N
0006- N1
0008- N2
0843- NDEC
09EA- NEW
091D- NORMIN
094D- NORMOT
0954- NORMOT1
095B- NORMOT2
0962- NORMOT3
09E6- OVER
097B- OVERFLOW
0B19- PRINTN
00F9- PROD
DB3A- PRTSTR
```

Tabella dei simboli del listato 1 (segue).

```
09F3- PT
00ED- REG1
09FD- RICH0
0A0B- RICH1
0A19- RICH2
0A29- RISP1
0A36- RISP2
09DD- RLQL
097D- SCAMBIO
0938- SEGN2
09F2- SEGNO
09D4- SRQL
0B01- STAMPA
0876- TWOCOMP
09A1- UNSDV1
09B2- UNSDV2
08FE- UNSM1
0916- UNSM2
08FB- UNSM0
```

```
5  HOME
10  INPUT "QUANTE VOLTE ? ";MAX
20  INPUT "PRIMO NUMERO ? ";N1
30  INPUT "SECONDO NUMERO ? ";N2
40  FOR I = 1 TO MAX
50 RIS = N1 / N2
60  NEXT I
70  PRINT "RISULTATO = ";RIS
80  END
```

Listato 2 *Applesoft (interpretato e compilato).*

```
10 PRINT "QUANTE VOLTE ";: INPUT MAX
20 PRINT "PRIMO NUMERO ";: INPUT N1
30 PRINT "SECONDO NUMERO ";: INPUT N2
40 FOR I=1 TO MAX
50 RIS=N1/N2
60 NEXT I
70 PRINT "RISULTATO = ";RIS
80 END
```

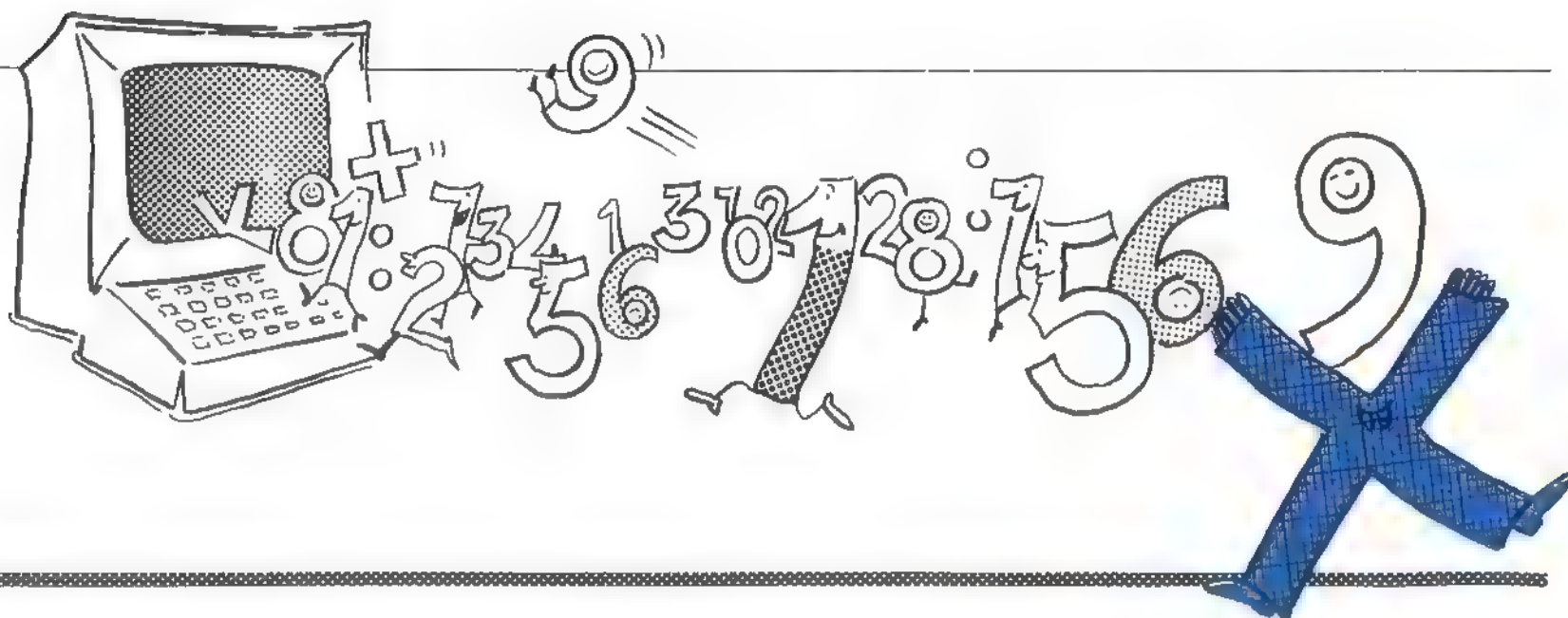
Listato 3. *Integer Basic.*

```
5  HOME
10  INPUT "QUANTE VOLTE ? ";MAX%
20  INPUT "PRIMO NUMERO ? ";N1%
30  INPUT "SECONDO NUMERO ? ";N2%
35 I% = 1
40  IF I% > MAX% THEN 70
50 RIS% = N1% / N2%
60 I% = I% + 1: GOTO 40
70  PRINT "RISULTATO = ";RIS%
80  END
```

Listato 4. *Mini-compilatore.*

```
PROGRAM TEST;
  VAR
    NUM, DEN, RIS, I, MAX : INTEGER;

BEGIN
  WRITE ('QUANTE VOLTE ? ');
  READLN (MAX);
  WRITE ('PRIMO NUMERO ? ');
  READLN (NUM);
  WRITE ('SECONDO NUMERO ? ');
  READLN (DEN);
  FOR I := 1 TO MAX DO
    RIS := NUM DIV DEN;
  WRITELN ('RISULTATO = ',RIS)
END.
```

Listato 5. *Pascal.*

A. funzione

Abbreviazione di ABS (vedi), di AT (vedi), di AND (vedi).

ABS funzione (ANSI)

La funzione ABS fornisce il valore assoluto del numero specificato. A=ABS(N) significa che A viene posto uguale al valore assoluto di N. Esempio: ABS(−12) è 12.

La funzione ABS è presente in tutti gli interpreti Basic. Talvolta è abbreviata con A.

AND operatore

L'operatore logico AND è utilizzato per connettere due condizioni logiche (vedi riquadro). Per esempio IF A=8 AND B>5 THEN RETURN significa che se il valore di A è 8 *e* il valore di B è maggiore di 5 deve essere eseguita l'istruzione RETURN. Quando due relazioni *x* e *y* sono connesse da un operatore AND, l'intera relazione *x* AND *y* è verificata (o vera) se sia *x* che *y* sono verificate (o vere), secondo il seguente schema

x	*y*	*x* AND *y*
vera	vera	vera
vera	falsa	falsa
falsa	vera	falsa
falsa	falsa	falsa

L'operatore AND è presente nella gran parte degli interpreti Basic. Talvolta è abbreviato con A., *, &.

Se il tuo computer non l'ha

In una istruzione IF-THEN l'operatore AND può essere simulato con due istruzioni IF-THEN. Per esempio IF A=8 AND B>5 THEN *z* si può scrivere

```
10 IF A< > 8 THEN 30
20 IF B > 5 THEN z
30 prossima istruzione
```

ACS funzione

La funzione ACS è utilizzata in alcuni interpreti Basic per calcolare l'arcocoseno del numero specificato in radianti. A=ACS(N) significa che A viene posto uguale all'arco (espresso in radianti) che ha N come coseno (il valore di N deve essere compreso tra −1 e 1). Per esempio, ACS(0) è 1.5708 radianti (π/2).

La funzione ACS non è presente in tutti gli interpreti Basic; talvolta è indicata con ARCOS (ZX80) o con AC. (TRS-80 Pocket Computer).

Se il tuo computer non l'ha

Può essere simulata mediante ATN (vedi) e SQR (vedi): ACS(X) si può scrivere

```
2*(ATN(1) − ATN (X/(1+SQR (1−X*X))))
```

(il valore di X deve essere compreso tra −1 e 1).

ASC funzione

La funzione ASC fornisce il codice ASCII decimale del carattere usato come argomento. A=ASC(A$) significa che A viene posto uguale al codice ASCII del primo carattere della stringa A$. Per esempio ASC("B") è 66.

Su Atom Acorn è indicata con CH.

La funzione inversa di ASC è CHR$ (vedi).

' (apostrofo) istruzione o operatore

L'apostrofo è usato da molti interpreti Basic come abbreviazione dell'istruzione REM (vedi).

Inizia questo mese il *Dizionario di Basic*, una guida alfabetica ragionata a tutte le istruzioni, le funzioni, gli operatori presenti negli interpreti Basic più diffusi, con indicazioni sugli usi particolari, le eccezioni, le modalità d'uso, la simulazione di istruzioni e funzioni non presenti.

Ogni mese presenteremo, in ordine alfabetico, una serie di schede illustrative e di riquadri su argomenti particolari (la grafica, il codice ASCII, le espressioni logiche, e così via).

Come sempre, chiediamo il contributo dei lettori per eventuali correzioni, aggiunte, suggerimenti. Scrivete a

Personal Software
Rubrica "Dizionario di Basic"
Via Rosellini 12
20124 MILANO

Per esempio

10' COMMENTO

Alcuni interpreti usano l'apostrofo per delimitare le stringhe (per questo scopo la maggior parte degli interpreti usa le virgolette (vedi)). Per esempio

10 PRINT'STRINGA'

Infine l'apostrofo è usato dal Basic Atom Acorn come indicatore di ritorno cursore (CR o RETURN) nelle istruzioni PRINT. Per esempio, se A=1, B=2, C=3, PRINT A'B'C stamperà

1
2
3

ASN funzione

La funzione ASN è utilizzata in alcuni interpreti Basic per calcolare l'arcoseno del numero specificato in radianti. A=ASN(N) significa che A viene posto uguale all'arco (espresso in radianti) che ha N come seno (il valore di N deve essere compreso tra −1 e 1). Per esempio ASN(0) è 0 radianti.

La fuzione ASN non è presente in tutti gli interpreti Basic; talvolta è indicata con ARCSIN (ZX80).

Se il tuo computer non l'ha

Può essere simulata mediante ATN (vedi) e SQR (vedi):

ASN(X) si può scrivere
2*ATN(X/(1+SQR(1−X*X)))

AT funzione

La funzione AT è utilizzata dopo una PRINT per indicare la posizione d'inizio della stampa. L'argomento di AT può essere un numero, una variabile numerica o una espressione numerica. Tra AT e l'argomento della PRINT deve essere inserita una virgola o un punto e virgola.

La funzione AT è utilizzata nel TRS-80 Level I Basic. Altri interpreti (Level II) usano @ per questo scopo.

ATN funzione (ANSI)

La funzione ATN calcola l'arcotangente di un numero specificato in radianti. A=ATN(X) significa che A viene posto uguale all'arco (espresso in radianti) che ha X come tangente. Per esempio ATN(2) è 1.10715 radianti.

Condizioni logiche

In Basic si può formare una *condizione logica*, detta anche *espressione relazionale*, confrontando espressioni numeriche (o di stringa) mediante gli operatori (detti di selezione)

= uguale
> maggiore
< minore
< > o # diverso
< = minore o uguale
> = maggiore o uguale

Ecco alcuni esempi:

(a) A > = 14.3
(b) A < B
(c) A+1 < > B*2−7
(d) A$ < B$
(e) A$ = "F"
(f) A$ < > "STOP"

La condizione logica così formata può essere *vera* o *falsa* per particolari valori delle variabili. Per esempio la condizione (a) è vera se A=20, è falsa se A=10, e analogamente per le altre. Più condizioni logiche si possono congiungere in un'unica condizione logica mediante gli operatori AND (vedi) e OR (vedi); una condizione logica si può negare mediante l'operatore NOT (vedi). Generalmente le condizioni logiche si usano nelle istruzioni IF-THEN (vedi) per permettere salti condizionati.

Alcuni computer tuttavia assegnano alle condizioni vere un certo valore e alle condizioni false un altro valore (generalmente scelti tra 0, 1 e −1) ed è possibile utilizzare direttamente tali valori, per esempio scrivendo

PRINT B=0

e ottenendo il valore corrispondente a "vero" se B è zero o quello corrispondente a "falso" se B è diverso da zero. Altro esempio:

A=B+(C<2)

dove ad A viene assegnato il valore di B più il valore corrispondente a "vero" se C è minore di 2, oppure il valore corrispondente a "falso" se C è maggiore o uguale a 2. Analogamente:

A=(L=100)

Questa istruzione assegnerà "vero" ad A se L è 100, "falso" se L è diverso da 100.

I valori assegnati dai principali interpreti Basic alle condizioni vere o false sono i seguenti.

	Applesoft e Integer Basic	Atom Acorn	Commodore	ZX80 ZX81
vero	1	1	−1	1
falso	0	0	0	0

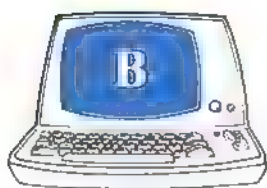
La funzione ATN è presente nella maggior parte degli interpreti Basic (non è presente in Integer Basic). Talvolta è indicata con ATAN, ARCTAN (ZX80).

Importanti utilizzazioni
Molti interpreti Basic hanno ATN come unica funzione circolare inversa (non hanno cioè ACS e ASN). Ciò è dovuto al fatto che è la più immediata da usare per trovare le altre funzioni circolari inverse, le cui formule in funzione di X sono
arcocoseno (vedi ACS)
2*(ATN(1)−ATN(X/(1+SQR(1−X*X))))
arcocotangente ATN (1/X)
arcosecante ATN (1/SQR (X*X−1))
arcocosecante ATN (SQR (X*X−1))
arcoseno (vedi ASN) 2*ATN (X/(1+SQR(1−X*X)))

Inoltre ATN è l'unico modo per calcolare immediatamente (e con la maggior precisione possibile) il valore di pi greco, quando non è presente come costante:

PI = 4*ATN(1)

AUTO comando

Il comando AUTO fornisce automaticamente i numeri di linea durante la battitura di un programma Basic. Per esempio AUTO 100,5 parte dal numero 100 e prosegue con incrementi di 5 (100, 105, 110, 115, ...)
Il comando AUTO è disponibile solo in alcuni interpreti (Integer Basic, TRS-80 Level II Basic). Negli altri può essere aggiunto come routine di utilità.

BREAK tasto

Il tasto BREAK interrompe l'esecuzione del programma. L'esecuzione può essere continuata con il comando CONT (vedi). Vedi anche STOP. Sul PET/CBM il tasto è indicato RUN/STOP e premendolo appare il numero di linea in cui il programma si è fermato.
Sul TRS-80 il tasto BREAK è presente sia in livello I che in livello II. Per disabilitarlo POKE 16396,23. Per riabilitarlo POKE 16396,201 o POKE 16396,20. Vedi POKE.
Sull'Apple il tasto è indicato RESET e su alcuni modelli bisogna premerlo assieme a CTRL.

C. comando

Abbreviazione di CONT (vedi).

CALL comando

CALL è un comando presente in qualche interprete Basic che permette di passare il controllo ad un programma scritto in linguaggio macchina. Per esempio CALL 18624 trasferisce il controllo al programma che inizia all'indirizzo 18624.
Alcuni interpreti usano USR (PET/CBM, TRS-80) (vedi). Vedi anche SYS.

CDBL funzione

La funzione CDBL viene utilizzata per cambiare numeri o variabili numeriche dalla singola precisione alla doppia precisione (C=*change*, DBL=*double*).
Nel TRS-80 Level II Basic, i numeri in singola precisione contengono 6 cifre significative, quelli in doppia precisione 16 cifre significative. Esempio:

```
10 CLS
20 X=6: Y=7
30 PRINT"IN SINGOLA PRECISIONE 6/7=";X/Y
40 PRINT"IN DOPPIA PRECISIONE 6/7=";
   CDBL(X)/CDBL(Y)
```

Si ottengono questi risultati:

```
IN SINGOLA PRECISIONE 6/7= .857143
IN DOPPIA PRECISIONE 6/7= .8571428571428571
```

Si noti che in singola precisione l'ultimo numero è arrotondato.

CH funzione

Nel Basic dell'Atom Acorn la funzione CH identifica il codice ASCII del primo carattere di una stringa. Per esempio PRINT CH"ACORN" stampa il numero 65 (il codice ASCII di A).
Vedi ASC. ■

Grafica tridimensionale con APPLE

Questo programma, per Apple II con 48 K, permette di tracciare disegni e di ruotarli, ingrandirli, ridurli, spostarli sullo schermo

di *Mark Pelczarski*

Le parole che state leggendo su queste pagine sono scritte su una superficie a due dimensioni, ma, se date un'occhiata intorno, vi accorgete subito che il mondo che ci circonda ne possiede una terza: la profondità. Pensate ora di osservare lo schermo di un computer: le immagini vi appaiono su una superficie bidimensionale. Eppure sullo stesso schermo potete assistere a spettacoli televisivi o film che danno la sensazione della profondità. Quando i personaggi sullo schermo si allontanano, la loro immagine diventa più piccola, mentre quando si avvicinano si ingrandisce.

Il programma presentato in questo articolo gira su un Apple II con 48 K di RAM, minifloppy e linguaggio Applesoft. Esso vi permette di tracciare disegni che potete ruotare, ingrandire, ridurre o muo-

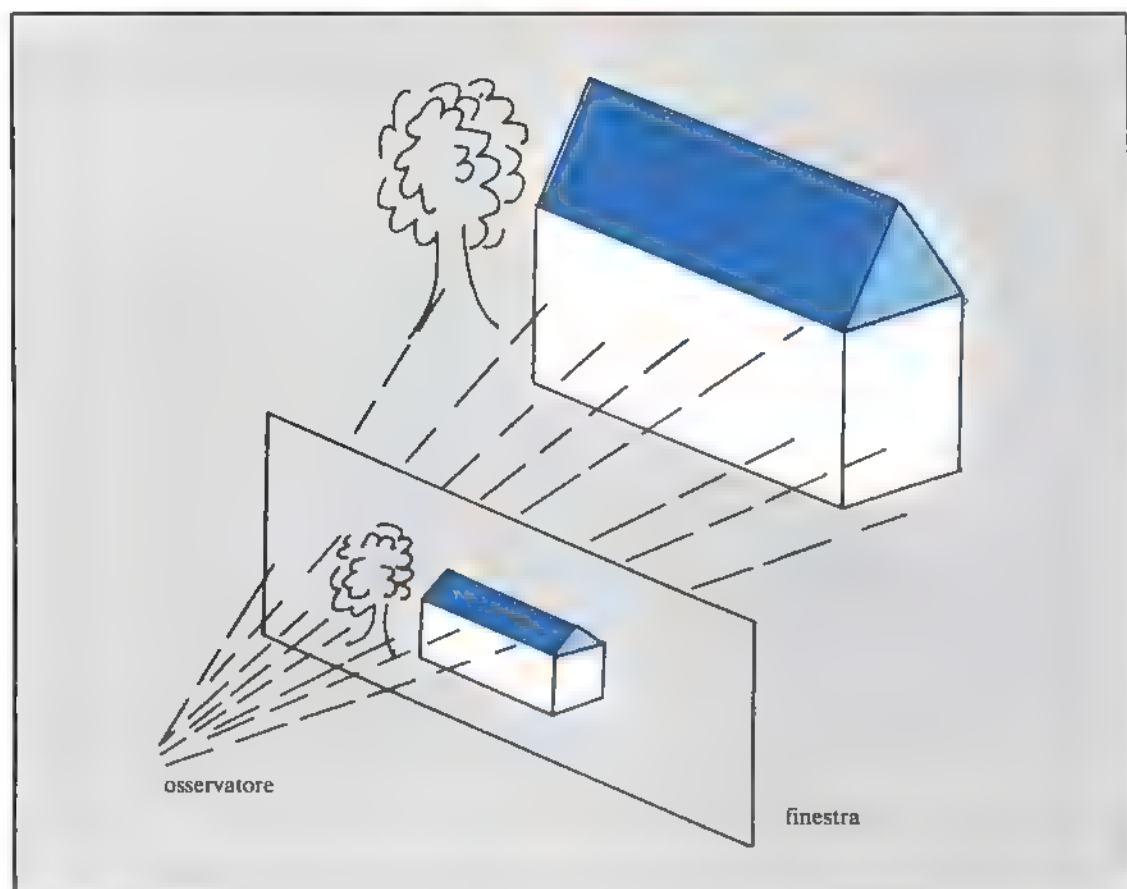



Figura 1. *Proiezione di un oggetto tridimensionale su una superficie.*

Questo programma è disponibile su disco. Vedete nelle ultime pagine il "Servizio Programmi".

Traduzione di *S. Ventura*

vere in quelle che sullo schermo appaiono come tre dimensioni. Il programma è stato scritto in Basic, quindi non aspettatevi di ottenere animazioni veloci; è però molto preciso e facile da usare.

Proiezione di immagini tridimensionali

Per cominciare, diamo un'occhiata alle tecniche di rappresentazione di oggetti tridimensionali in uno spazio a due dimensioni, quale è uno schermo televisivo, senza curarci eccessivamente degli aspetti matematici del problema, almeno per il momento. Pensate allo schermo del vostro monitor come ad una finestra, e ad alcuni oggetti tridimensionali reali al di là del vetro. Meglio ancora, trovatevi una finestra e una matita grassa. Sedetevi abbastanza vicini da poter raggiungere il vetro e disegnate il contorno di quello che vedete fuori dalla finestra con la matita (assicuratevi, però, di poter cancellare in seguito quello che disegnerete). Dovrete trovarvi, alla fine, con una rappresentazione a due dimensioni (la superficie della finestra) della vista esterna; dovrebbe anche essere gradevolmente accurata, o, come si dice nel gergo della grafica 3-D, in "prospettiva reale".

Come funziona il procedimento, e come si può tradurlo in un programma per computer? Osservate il disegno in figura 1. La luce viaggia in linea retta dall'oggetto reale, attraverso la finestra, fino ai vostri occhi (si trascura la rifrazione nel vetro). Dove le linee incontrano la finestra si ottiene il profilo del mondo all'esterno, proiettato su una superficie a due dimensioni. La stessa cosa accade sulla pellicola di una macchina fotografica. La chiave matematica sta nel fatto che ci sono dei punti da una parte di un piano, congiunti con rette a un solo punto dalla parte opposta del piano, e, dove queste linee intersecano il piano, si forma la proiezione bidimensionale. I punti dalla prima parte sono gli oggetti, mentre il punto dall'altra parte rappresenta i vostri occhi, il piano è la finestra e le linee sono la luce.

Assegnamento della memoria

Ci servono alcune strutture per rendere eseguibile col computer il procedimento appena visto. Definiremo i nostri oggetti come figure formate da segmenti. Memorizzeremo un gruppo di punti come coordinate tridimensionali: X, Y e Z. Per restare vicini a quello che già sapete della grafica sullo schermo, X sarà la misura, da sinistra a destra dello schermo, della larghezza, Y misurerà l'altezza dal basso verso l'alto e Z rappresenterà la profondità, dalla superficie dello schermo verso il retro dell'apparecchio (vedi figura 2). Il

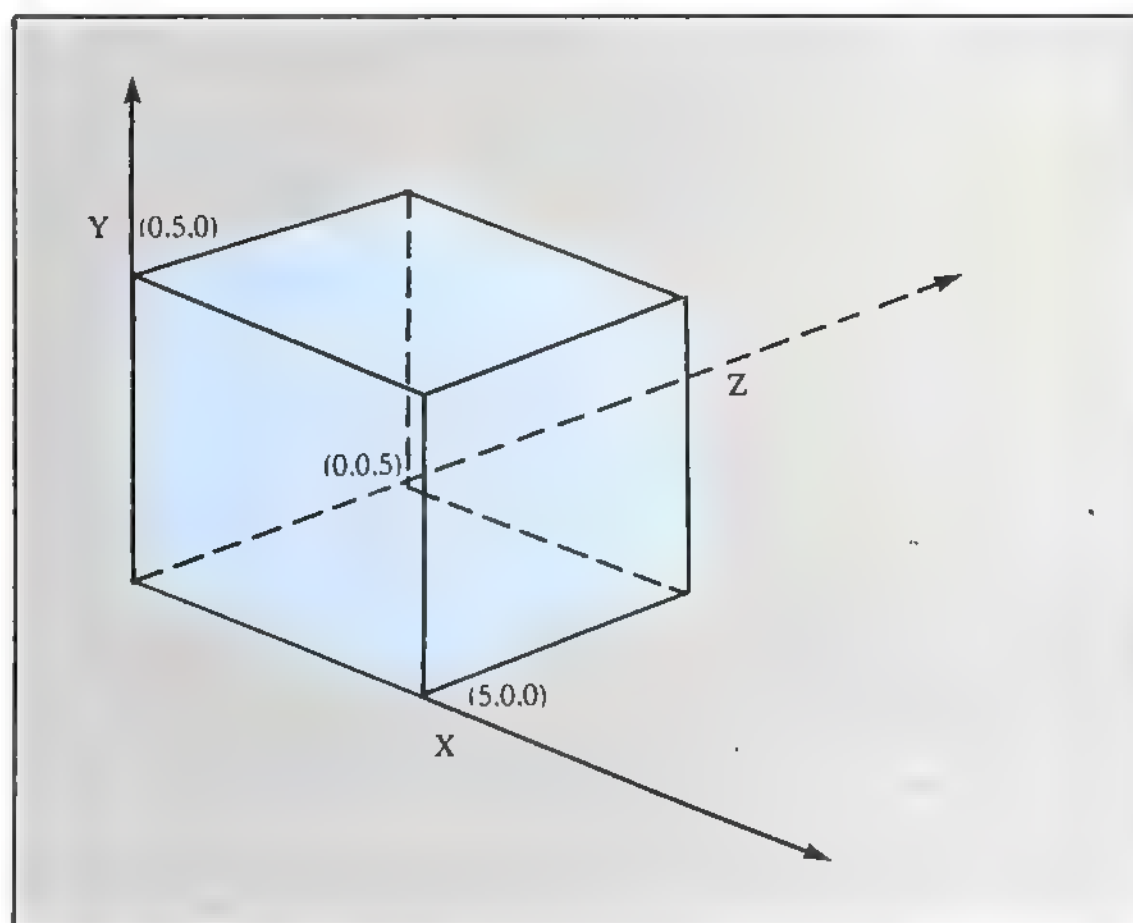



Figura 2. *Assi.*

Punti				Linee		
#	X	Y	Z	#	Da	a
1	0	0	0	1	0	1
2	5	0	0	2	1	2
3	5	5	0	3	2	3
4	0	5	0	4	3	0
5	0	0	5	5	4	5
6	5	0	5	6	5	6
7	5	5	5	7	6	7
8	0	5	5	8	7	4
				9	0	4
				10	1	5
				11	2	6
				12	3	7

Figura 3. *Punti e linee per il cubo di figura 2.*

punto (0,0,0) sarà il centro dello schermo. (Chi è pratico di rappresentazioni in coordinate 3-D noterà che gli assi sono ruotati di 90° indietro rispetto alla normale orientazione, per poter definire Z come profondità.)

I punti che vengono memorizzati sono semplicemente gli estremi dei segmenti. Non è necessario memorizzare ciascun punto del segmento, poiché in proiezione le linee congiungeranno ancora i loro rispettivi estremi. Oltre alle coordinate dei punti verrà memorizzata anche una lista di linee. Queste saranno identificate dai loro estremi, una specie di connessione da punto a punto in 3-D. La figura 3 mostra come viene memorizzato il cubo di figura 2.

In un Apple con 48 K di memoria c'è abbastanza posto per memorizzare comodamente 500 punti e 750 linee, definiamo, perciò, lo spazio di memoria in questo modo:

- X(499) coordinata *x* di ogni punto da 0 a 499
- Y(499) coordinata *y* di ogni punto
- Z(499) coordinata *z* di ogni punto
- L%(749,1) estremi delle linee da 0 a 749; L%(I,0) è un estremo della riga *i*-esima, L%(I,1) è l'altro estremo.

Il segno "%" rende L% una variabile intera, che occupa 2 byte per elemento anziché 5 come una variabile in virgola mobile.

Le coordinate di uno degli estremi della linea *i*-esima si trovano in questo modo:

X(L%(I,0)), Y(L%(I,0)), Z(L%(I,0))

dove I è il numero della linea e L%(I,0) contiene l'indice del punto.

In realtà il programma usa la matrice P(499,2) per memorizzare i punti, piuttosto che X,Y e Z. Se N è il numero di un punto, P(N,0) è la coordinata *x*, P(N,1) è la coordinata *y* e P(N,2) è la coordinata *z*. Questo accorcia parti del programma permettendo l'uso di cicli, ma nel corso di questo articolo useremo X, Y e Z.

Mettiamo un oggetto sullo schermo

Tutti gli elementi necessari sono là: i punti sono memorizzati, lo schermo è il piano *xy*, e i vostri occhi sono in qualche luogo al di fuori, dalla parte negativa dell'asse *z* (nel programma, D1 rappresenta questa distanza). Quando ho cominciato questa parte del programma, pensavo che ci fosse qualche grossa difficoltà matematica implicita. Dopo giorni di consultazione di vecchi testi di matematica, tra equazioni a tre dimensioni, equazioni dei piani, tecniche per trovare l'intersezione tra linee e piani, formule per trovare i punti proiettati, arrivai ad una relazione piuttosto semplice: la proporzione. Avete bisogno delle coordinate X e Y sullo schermo e avete a disposizione le coordinate X, Y e Z del punto che volete proiettare. Le linee mostrate in figura 4 formano due triangoli simili. Potete calcolare X e Y separatamente; la figura e le formule che seguono mostrano come calcolare Y.

$$\frac{Y}{D1 + Z} = \frac{\text{proiezione di Y}}{D1}$$

da cui

$$\text{proiezione di Y} = \frac{Y*D1}{D1+Z}$$

Questi calcoli vengono eseguiti nelle righe 4385 e 4390 del programma. Qui però ho già cambiato D1 in VZ, che compie alcuni cambi di scala per rendere le dimensioni compatibili con quelle dello schermo. TR è il punto traslato sullo schermo. Fatto questo con le coordinate X e Y di ciascun estremo, viene disegnata una linea che connette i punti traslati. Il procedimento viene ripetuto per ogni gruppo di punti di ogni linea.

Divertiamoci muovendo gli oggetti

Guardare solamente un oggetto tridimensionale sullo schermo non è granché eccitante se non potete farci qualcosa, come muoverlo o girarlo per vederlo da un'altra angolazione. È più divertente poter vedere sullo schermo qualcosa che assomigli ad esempio alla macchina sportiva che sta girando l'angolo dalla finestra. Prima ne vedete il fianco, poi il muso che gira, quindi diventa più grande (o meglio sembra) mentre si avvicina, e prosegue la sua corsa.

Ci sono due modi di vedere altre angolazioni di un oggetto: muovere l'oggetto o spostare il punto di vista. Muovere l'oggetto implica il cambiamento di tutte le coordinate

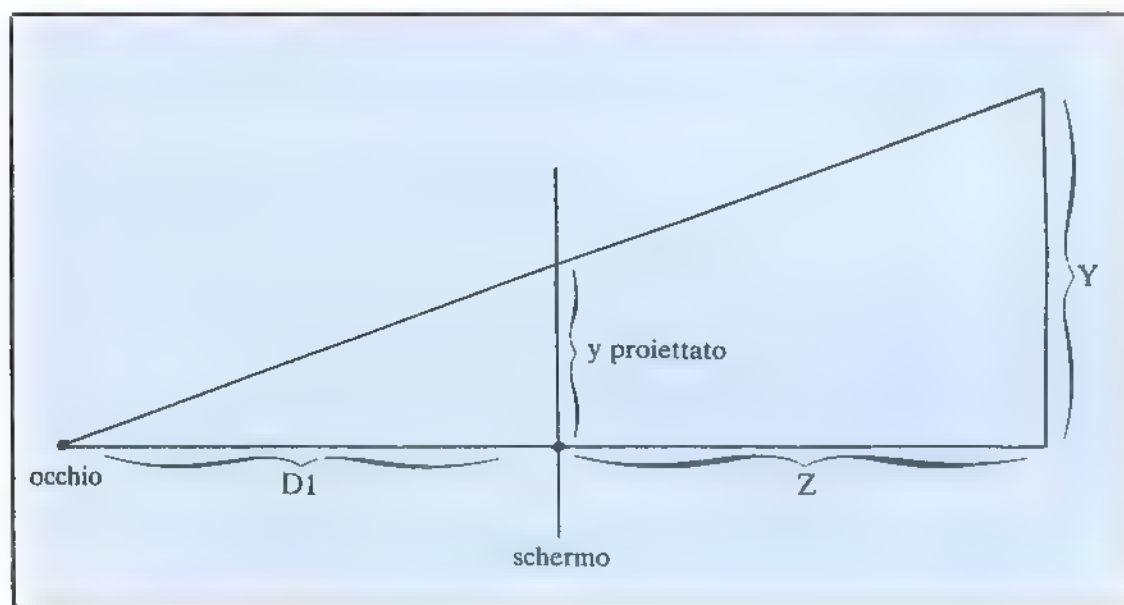



Figura 4. *Proiezione di y sullo schermo.*

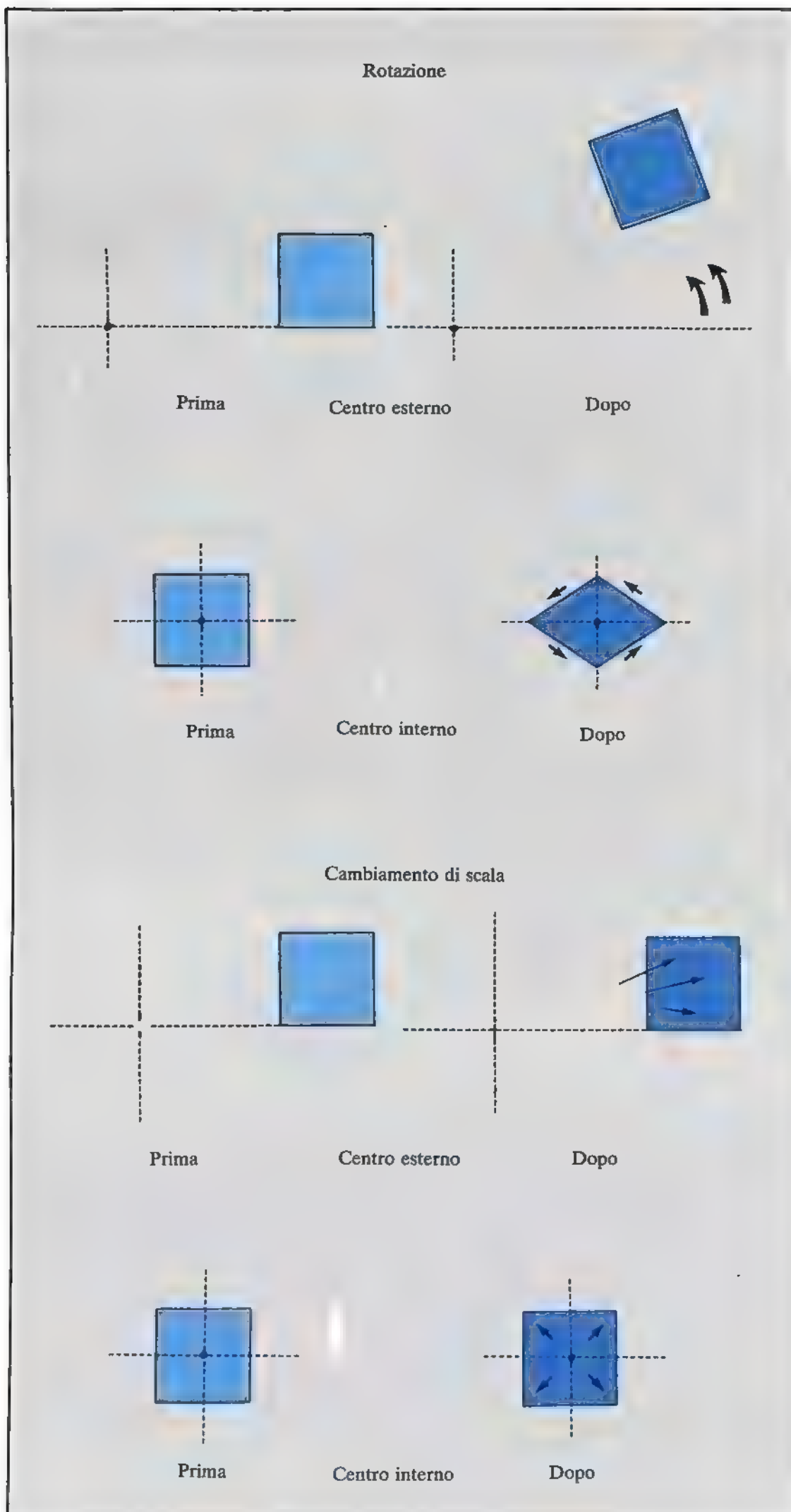



Figura 5. *Rotazione e cambiamento di scala con centri all'interno e all'esterno dell'oggetto.*

che lo compongono. Muovere il punto di vista sembra richiedere il cambiamento di un solo punto, ma rende la traslazione in due dimensioni un po' più lunga. Con questo programma è meglio spostare l'oggetto, poiché, in seguito, tratteremo il problema della presenza contemporanea di più di un oggetto sullo schermo e del loro movimento indipendente (come, per esempio, muovere una scatola che sta sopra ad un'altra).

Esistono tre tipi di operazioni che possiamo compiere su un oggetto: rotazione, spostamento e cambio di scala. Ogni operazione influisce in qualche modo sulle coordinate di ogni punto memorizzato. Nessuna di loro, invece, ha effetto sulla lista delle linee, visto che il loro scopo è semplicemente di congiungere i loro estremi.

Spostamenti

Spostare un oggetto significa muoverlo in una direzione ed è l'operazione più semplice da fare. Possiamo spostare un oggetto a sinistra o a destra semplicemente sommando un numero positivo o negativo, rispettivamente, a ogni sua coordinata x. Per spostare in su o in giù si aggiunge alla coordinata y di ogni punto il valore dello spostamento e per spostarlo in avanti o indietro (verso o lontano da voi) si deve modificare la coordinata z. L'effetto sullo schermo quando un oggetto viene spostato a destra, a sinistra, in su o in giù è di muovere l'oggetto in quella direzione, ma anche di aumentare o diminuire la porzione laterale vista. Si può paragonare alla vista che si ha di un palazzo proprio di fronte oppure da un po' più lontano di lato sulla strada. Da lontano non vedete solo la facciata, ma anche una parte del lato. Spostare un oggetto avanti e indietro lo farà apparire più grande o più piccolo, come accade con gli oggetti reali quando state più vicini o lontani da essi.

Cambiamento di scala

La rotazione e il cambiamento di scala pongono un nuovo problema: entrambe richiedono un punto di riferimento. Nel cambio di scala esiste un punto che rimane inalterato. Per la rotazione, invece, avete bisogno di un punto attorno al quale far ruotare l'oggetto. In entrambi i casi è opportuno che questo punto coincida con il centro dell'oggetto. La figura 5 mostra degli esempi di entrambe le operazioni, ognuna fatta prima con il punto di riferimento all'esterno dell'oggetto, poi all'interno. Nella maggior parte dei casi se il punto si trova all'esterno dell'oggetto l'operazione provoca la fuoriuscita dell'oggetto dallo schermo. Per risolvere questo problema, prima di ogni altra operazione il programma calcola il punto centrale di ogni figura. Ciò avviene nelle righe 3032 - 3038 calcolando la media tra il più piccolo e il più grande valore della coordinata *x*, poi ripetendo i calcoli per le altre due coordinate; le variabili CR(0), CR(1) e CR(2) contengono i valori delle coordinate X, Y e Z del centro così calcolato.

Per cambiare scala ad un oggetto, il passaggio principale consiste nel moltiplicare ogni coordinata per un fattore di scala, per esempio 2 per raddoppiarne le dimensioni. Eseguendo questa operazione per il cubo di figura 2 e figura 3, tutti i 5 vengono trasformati in 10, raddoppiando la lunghezza dei lati. Senza far riferimento a un centro, tuttavia, è possibile anche far sparire gli oggetti, cioè farli uscire dallo schermo. Il centro apparente per una moltiplicazione diretta è il punto (0,0,0). Per incorporare il vostro centro (quello calcolato) nell'operazione di cambio scala dovete:

1 Sottrarre le coordinate del centro da ogni punto. Questo trasforma la figura in un'altra identica che ha il centro in (0,0,0).

2 Moltiplicare ogni coordinata per la costante di scala. Questa operazione cambia la scala della figura con centro in (0,0,0), ora al suo interno.

Listato 1.

```
1       REM GRAFICI 3-D
5       LOMEM: 16384: HOME: D$=CHR$(4): HGR
50      DIM CR(2),[illegible](2),X(1),Y(1),P(499,2),L[illegible]
        TR(499,1),FG%(99,3),FT$(99)
65      GOSUB 350
70      HOME: VTAB 21: PRINT"1-[illegible]
        RE 3-VEDERE        4-RIPARTIRE   5-[illegible]
        -CARICARE      7-SALVARE   [illegible]
80      INPUT [illegible]
81      IF C=1 THEN [illegible]
82      IF C=8 THEN TEXT: [illegible]
83      IF NF [illegible] THEN [illegible]
84      IF C[illegible]1 AND C[illegible]6 THEN PRINT: PRINT
        "NON CI SONO FIGURE IN MEMORIA[illegible]": PRINT
        "BATTI UN TASTO "[illegible]: GET [illegible]: GOTO [illegible]
85      ON C GOSUB 5,1[illegible]
86      GOTO 70
120     HGR: [illegible]=1: GOSUB [illegible]910
130     GOSUB 4000: SW=1: GOSUB [illegible]
155     IF ES=1 THEN SW=2: GOSUB [illegible]
165     GOTO 1[illegible]
170     GOSUB 2370: C1[illegible]
175     TEXT: HOME: PRINT FT$(CF[illegible]
180     PRINT"1-PUNTI, 2-LINEE[illegible]
        INPUT C: IF C[illegible] OR [illegible] THEN 18[illegible]
185     ON [illegible] GOTO 190,205,[illegible]
190     PRINT"#          X[illegible]
        FOR I=[illegible]
194     PRINT I-C1+1;: [illegible]
        PRINT LEFT$([illegible]
        S1=S1+1: IF S1=[illegible] THEN PRINT
        "BATTI UN TASTO ";: GET A$: [illegible]: PRINT
196     NEXT: GOTO 180
205     C=FG%(CF,2): PRINT"#","DA"[illegible]
        FOR I=C TO FG%(CF,3)
212     PRINT I-C+1,L%(I,0)-C1+1,L%(I,1)-C1+1:
        S1=S1+1: IF S1=20 THEN PRINT
        "BATTI UN TASTO ";: GET A$: [illegible]: PRINT
214     NEXT: GOTO 180
220     IF SW=2 THEN 230
222     INPUT"PUNTO #";I: I=I+C1-1:
        IF I<C1 OR I>FG%(CF,1) THEN 180
224     INPUT"X:";P(I,0): INPUT"Y:";P(I,1): INPUT
        "Z:";P(I,2): GOTO 180
230     C=FG%(CF,2): INPUT"LINEA #";I: I=I+C-1:
        IF I<C OR I>FG%(CF,3) THEN 180
232     INPUT"DAL #";L%(I,0): INPUT"AL #";L%(I,1):
        FOR I1=0 TO 1: L%(I,I1)=L%(I,I1)+C1-1: NEXT:
        GOTO 180
239     RETURN
250     INPUT"CON CHE NOME? ";A$
255     PRINT D$;"OPEN";A$
260     PRINT D$;"WRITE";A$
261     PRINT NP: PRINT NL: PRINT NF:
        IF NF<2 THEN 270
262     FOR I=0 TO NF-1: PRINT FT$(I):
        FOR I1=0 TO 3: PRINT FG%(I,I1): NEXT I1,I
270     FOR I=0 TO NP-1: FOR I1=0 TO 2:
        PRINT P(I,I1): NEXT I1,I
280     FOR I=0 TO NL-1: PRINT L%(I,0):
        PRINT L%(I,1): NEXT
```

(segue)

La terza possibilità, rotazione attorno all'asse *x*, produce un movimento alto/basso dell'oggetto. In questo caso è la coordinata *x* che non varia. Le formule sono:

nuovo Y = cos *a**Y − sen *a**Z
nuovo Z = cos *a**Z + sen *a**Y

Se studiate queste equazioni, noterete come i segni + e − dipendano dalla direzione cui assegnate angoli positivi. Ciò dipende dal fatto che l'opposto di un angolo ha lo stesso coseno dell'angolo considerato (cos(−*a*)=cos *a*) mentre i seni hanno valore opposto (sen(−*a*)=−sen *a*). Nel programma vengono assegnati valori negativi agli angoli che vanno in giù, a sinistra e in senso orario, mentre hanno valori positivi gli angoli che vanno in su, a destra e in senso antiorario. Tutto ciò viene gestito internamente; l'utilizzatore deve specificare solo la direzione (righe 6075 - 6110).

Prima di una rotazione, dobbiamo effettuare sul centro la stessa operazione fatta per il cambio di scala, altrimenti la rotazione farà girare l'oggetto intorno agli assi principali dello schermo, anziché su se stesso. La prima cosa da fare è sottrarre le coordinate del centro da tutte le altre coordinate dell'oggetto. A questo punto si possono applicare le formule trigonometriche appropriate per ruotare l'oggetto attorno a uno degli assi. Per ultima cosa si devono sommare le coordinate del vecchio centro alle coordinate dell'oggetto ruotato per riportarlo nella posizione originale.

Distorsioni

C'è un'ulteriore operazione nel programma chiamata distorsione. Una distorsione consiste nel cambiare scala all'oggetto in una dimensione: larghezza, altezza o profondità (rispettivamente coordinate X, Y o Z). L'effetto prodotto consiste nell'allungare o accorciare la figura in quella dimensione. Partendo da un cubo, per esempio, potete distorcere ciascuna dimensione, ottenendo un parallelepipe-

Segue listato 1.

```
2830   IF NF=1 THEN CF=0: RETURN
2845   INPUT"QUALE FIGURA? ";A$: I=0
2855   IF FT$(I)=A$ THEN 2870
2860   I=I+1: IF I<NF THEN 2855
2862   PRINT"NON NE HAI NESSUNA CHIAMATA ";A$:
       PRINT"<BATTI UN TASTO>";: GET A$: POP:
       RETURN
2870   CF=I
2875   RETURN
2900   IF NF<2 THEN C=1: GOTO 2910
2905   INPUT"1-TUTTE O 2-UNA? ";C
2910   IF C=1 THEN FS=0: SP=0: EP=NP-1: SL=0:
       EL=NL-1: GOTO 3032
2920   IF C<>2 THEN 2900
2930   GOSUB 2830: FS=1: SP=FG%(CF,0):
       EP=FG%(CF,1): SL=FG%(CF,2): EL=FG%(CF,3)
3032   FOR I=0 TO 2: CR(I)=999: T(I)=-999: NEXT
3033   FOR I=SP TO EP
3034   FOR I1=0 TO 2
3035   IF P(I,I1)<CR(I1) THEN CR(I1)=P(I,I1)
3036   IF P(I,I1)>T(I1) THEN T(I1)=P(I,I1)
3037   NEXT I1,I
3038   FOR I=0 TO 2: CR(I)=(CR(I)+T(I))/2: NEXT
3048   IF VS=1 THEN 3140
3049   VS=1: D1=0
3050   FOR I=SP TO EP
3060   VZ=0: FOR I1=0 TO 2:
       VZ=VZ+(CR(I1)-P(I,I1))^2: NEXT: VZ=SQR(VZ)
3110   IF VZ>D1 THEN D1=VZ
3120   NEXT
3130   VZ=-20*D1
3140   C=4: RETURN
4000   FOR I=SP TO EP
4101   IF C=4 THEN 4380
4102   FOR I1=0 TO 2: P(I,I1)=P(I,I1)-CR(I1):
       T(I1)=P(I,I1): NEXT
4110   ON C GOTO 4130,4200,4280,4380,4300
4130   T(1)=C1*P(I,1)-S1*P(I,2):
       T(2)=C1*P(I,2)+S1*P(I,1): GOTO 4350
4200   T(0)=C1*P(I,0)-S1*P(I,2):
       T(2)=C1*P(I,2)+S1*P(I,0): GOTO 4350
4280   T(0)=C1*P(I,0)-S1*P(I,1):
       T(1)=C1*P(I,1)+S1*P(I,0): GOTO 4350
4300   IF S1<>0 THEN T(S1-1)=P(I,S1-1)*M:
       GOTO 4350
4305   FOR I1=0 TO 2: T(I1)=P(I,I1)*M: NEXT
4350   FOR I1=0 TO 2: P(I,I1)=T(I1)+CR(I1): NEXT
4380   IF VZ-P(I,2)>-.001 THEN K=10000*D1:
       GOTO 4390
4385   K=VZ/(VZ-P(I,2))
4390   TR(I,0)=K*P(I,0): TR(I,1)=K*P(I,1)
4400   NEXT: RETURN
5000   IF SW=0 THEN HCOLOR=0: GOTO 5010
5005   IF FS=0 THEN HGR
5006   HCOLOR=7
5010   FOR I=SL TO EL
5020   SW=0
5030   FOR I1=0 TO 1
5035   IF L%(I,I1)<0 OR L%(I,I1)>=NP THEN SW=1:
       GOTO 5060
5040   X(I1)=TR(L%(I,I1),0)*CT:
```

(segue)

3 Sommare le coordinate del centro della figura originale ad ogni coordinata della figura. Con ciò il centro della figura modificata torna nella posizione di partenza, ma la figura è stata modificata esternamente ad esso.

Rotazioni

Le rotazioni, come i cambi di scala, hanno bisogno di un centro. Quando parliamo di direzione della rotazione, ci riferiamo a quella della parte dell'oggetto che vi è più vicina (quando la parte frontale gira verso sinistra, ad esempio, il retro gira a destra; chiamiamo questa la "rotazione a sinistra"). La rotazione può avvenire intorno a uno qualsiasi dei tre assi. Ruotando attorno all'asse y l'oggetto gira a sinistra o a destra, come una porta che si apre. La rotazione attorno all'asse z muove l'oggetto in senso orario o in senso antiorario, come le lancette dell'orologio.

In ogni caso, se come esempio prendiamo una rotazione attorno all'asse z, tutte le coordinate z rimangono le stesse (rotazione in senso orario e antiorario non fa differenza: la profondità di ogni punto rimane inalterata). Potete pensare che il vostro oggetto sia bidimensionale, visto che le coordinate z non vengono coinvolte. Le coordinate x e y variano secondo le regole classiche della trigonometria. Le formule hanno a che fare con il seno e il coseno dell'angolo di rotazione a e sono le seguenti:

nuovo X = cos a*X = sen a*Y
nuovo Y = cos a*Y + sen a*X
(nuovo Z = Z)

Poiché la rotazione avviene intorno all'asse z, la figura sullo schermo ruoterà in senso orario o antiorario, a seconda dell'angolo. Un angolo positivo provoca una rotazione antioraria.

Allo stesso modo, le rotazioni intorno all'asse y (destra/sinistra) non hanno effetto sulle coordinate y (altezze degli oggetti sullo schermo). Le relative formule sono:

nuovo X = cos a*X − sen a*Z
nuovo Z = cos a*Z + sen a*X

Segue listato 1.

```
286   PRINT D$;"CLOSE";A$
288   RETURN
300   ONERR GOTO 302
301   INPUT"CHE NOME?";A$: GOTO 304
302   PRINT A$;" NON E' SUL DISCO": PRINT
      "<BATTI UN TASTO>";: GET A$: POKE 216,0:
      GOTO 70
304   PRINT"VUOI TENERE ";A$;" COME NOME";:
      INPUT FT$(NF)
305   IF LEFT$(FT$(NF),1)="S" THEN FT$(NF)=A$:
      GOTO 308
306   IF LEFT$(FT$(NF),1)<>"N" THEN 304
307   INPUT"NUOVO NOME?";FT$(NF)
308   PRINT: PRINT D$;"OPEN";A$
309   PRINT D$;"READ";A$
310   INPUT T(0): INPUT T(1): INPUT T(2)
311   IF T(2)<2 THEN 321
312   FOR I=NF+1 TO NF+T(2)
313   INPUT FT$(I)
314   FOR I1=0 TO 1: INPUT FG%(I,I1):
      FG%(I,I1)=FG%(I,I1)+NP: NEXT
317   FOR I1=2 TO 3: INPUT FG%(I,I1):
      FG%(I,I1)=FG%(I,I1)+NL: NEXT I1,I
321   FOR I=NP TO NP+T(0)-1: FOR I1=0 TO 2:
      INPUT P(I,I1): NEXT I1,I
325   FOR I=NL TO NL+T(1)-1: FOR I1=0 TO 1:
      INPUT L%(I,I1): L%(I,I1)=L%(I,I1)+NP:
      NEXT I1,I
331   FG%(NF,0)=NP: FG%(NF,1)=NP+T(0)-1:
      NP=NP+T(0): FG%(NF,2)=NL:
      FG%(NF,3)=NL+T(1)-1: NL=NL+T(1): CF=NF:
      NF=NF+T(2)+1: IF T(2)=1 THEN NF=NF-1
334   PRINT D$;"CLOSE";A$
336   POKE 216,0: RETURN
350   NL=0: NP=0: NF=0: VS=0: CT=3: RETURN
400   INPUT"CON CHE NOME? ";A$
410   PRINT D$;"BSAVE";A$;",A8192,L8192"
420   RETURN
1000  HOME: TEXT: CF=NF: NF=NF+1: INPUT
      "NOME DELLA FIGURA? ";FT$(CF): FG%(CF,0)=NP:
      FG%(CF,2)=NL: PRINT
      "BATTI 'F' QUANDO NON CI SONO PIU' PUNTI.":
      ONERR GOTO 1010
1010  PRINT"PUNTO #";NP-FG%(CF,0)+1: INPUT"X:";A$:
      IF LEFT$(A$,1)="F" THEN FG%(CF,1)=NP-1:
      GOTO 2000
1015  IF ASC(A$)>57 THEN 1010
1020  P(NP,0)=VAL(A$): INPUT"Y:";P(NP,1): INPUT
      "Z:";P(NP,2): NP=NP+1: GOTO 1010
2000  PRINT
      "BATTI 'F' QUANDO NON CI SONO PIU' LINEE.":
      ONERR GOTO 2010
2010  PRINT"LINEA #";NL-FG%(CF,2)+1: INPUT
      "DAL PUNTO #";A$:
      IF LEFT$(A$,1)="F" THEN FG%(CF,3)=NL-1:
      POKE 216,0: GOTO 70
2015  IF ASC(A$)>57 THEN 2010
2020  L%(NL,0)=VAL(A$):
      INPUT"AL PUNTO #";L%(NL,1): FOR I=0 TO 1:
      L%(NL,I)=L%(NL,I)+FG%(CF,0)-1: NEXT:
      NL=NL+1: GOTO 2010
```

(segue)

do di qualsiasi misura. Grazie a questa operazione, potete creare, con poche figure di base, una infinita varietà di forme senza doverle definire una ad una.

Progetto del programma

Le principali opzioni in questo programma permettono di creare e editare le figure, osservarle e manipolarle, salvarle per usi futuri e caricarne di già esistenti. Altre opzioni incluse nel programma comprendono la capacità di cancellare tutte le figure dalla memoria per ripartire da capo, e di salvare su disco le immagini bidimensionali che appaiono sullo schermo.

È prevista anche la possibilità di avere più di una figura contemporaneamente in memoria. Ciò è stato possibile facendo in modo che parecchie piccole figure possano essere create, caricate da disco e manipolate separatamente, componendo una figura complessa, costituita da tutte le piccole figure memorizzate.

Questa figura può essere salvata, con tutte le piccole figure come suoi particolari. Le informazioni sulle figure piccole sono mantenute intatte, così quando viene richiamata la figura grande, quelle piccole possono ancora essere manipolate indipendentemente.

Per permettere questa capacità, sono stati usati altri due vettori supplementari: uno per i nomi delle figure in memoria e l'altro che contiene le informazioni sui primi e ultimi numeri dei punti e le prime e ultime righe di ogni figura. Il vettore dei nomi è FT$ e permette di memorizzare fino a cento nomi (0 - 99). La matrice delle informazioni è dimensionata FG% (99,3). Il 99 permette di memorizzare fino a 100 figure. Se I è il numero della figura, FG%(I,0) è il punto di partenza della figura, FG%(I,1) è il punto finale, FG%(I,2) è la prima linea e FG%(I,3) è l'ultima. Come esempio se la figura A avesse 8 punti (0-7) e 12 linee (0-11), e se la fig. B avesse 4 punti (8-11) e 4 linee (12-15), il punto di partenza di

Segue listato 1.

```
      Y(I1)=TR(L%(I,I1),1)*CT
5060  NEXT
5070  FOR I1=0 TO 1
5090  IF SW=1 THEN 5270
5100  IF ABS(X(I1))<=139 THEN 5190
5110  IF ABS(Y(I1))<=95 THEN 5150
5120  IF Y(0)=Y(1) THEN 5230
5125  YC=SGN(Y(I1))*95:
      XC=(YC-Y(1))*(X(0)-X(1))/(Y(0)-Y(1))+X(1):
      IF ABS(XC)<=139 THEN 5250
5150  IF X(0)=X(1) THEN 5230
5155  XC=SGN(X(I1))*139:
      YC=(XC-X(1))*(Y(0)-Y(1))/(X(0)-X(1))+Y(1):
      IF ABS(YC)<=95 THEN 5230
5180  GOTO 5230
5190  IF ABS(Y(I1))<=95 THEN 5270
5200  IF Y(0)=Y(1) THEN 5230
5205  YC=SGN(Y(I1))*95:
      XC=(YC-Y(1))*(X(0)-X(1))/(Y(0)-Y(1))+X(1):
      IF ABS(XC)<=139 THEN 5250
5230  SW=1: GOTO 5270
5250  X(I1)=XC: Y(I1)=YC
5270  NEXT
5280  IF SW=0 THEN HPLOT 140+X(0),96-Y(0) TO 140+
      X(1),96-Y(1)
5290  NEXT: RETURN
6000  HOME: VTAB 21: PRINT"1-RUOTA      2-SPOSTA
       3-SCALA OGGETTO 4-DISTORCE  5-MUOVE      6-C
      ENTRO         7-MENU       8-SCHERMO    ";
6038  IF FS=0 THEN PRINT"9-SCALA SCHERMO ";
6040  INPUT C: IF FS=0 AND C=9 THEN 6300
6050  ON C GOTO 6075,6142,6073,6071,6065,6200,
      6070,6250
6060  GOTO 6000
6065  GOSUB 2900: GOTO 6000
6070  POP: RETURN
6071  PRINT: INPUT
      "1-LARGHEZZA  2-ALTEZZA  3-PROFONDITA'";S1:
      IF S1<1 OR S1>3 THEN 6071
6073  IF C=3 THEN S1=0
6074  INPUT"MOLTIPLICARE PER? ";M: C=5: RETURN
6075  HOME: VTAB 21: PRINT"RUOTARE:": PRINT"1-IN B
      ASSO   2-IN ALTO   3-A SINISTRA   4-A DESTRA
         5-ORARIO    6-ANTIORARIO";: INPUT C:
      IF C<1 OR C>6 THEN 6075
6090  INPUT"ANGOLO (0 - 180) ? ";AN:
      IF AN<0 OR AN>180 THEN 6090
6110  AN=3.1415*AN/180:
      IF INT(C/2)*2<>C THEN AN=-AN
6130  S1=SIN(AN): C1=COS(AN): C=INT((C+1)/2):
      RETURN
6142  HOME: VTAB 21: PRINT"SPOSTARE:": PRINT"1-A S
      INISTRA   2-A DESTRA    3-IN GIU'    4-IN SU
          5-VICINO       6-LONTANO";: INPUT C:
      IF C<1 OR C>6 THEN 6142
6150  INPUT"DI QUANTE UNITA'? ";AN:
      IF INT(C/2)*2<>C THEN AN=-AN
6170  C=INT((C-1)/2): CR(C)=CR(C)+AN:
      FOR I=SP TO EP: P(I,C)=P(I,C)+AN: NEXT: C=4:
      RETURN
6200  PRINT"PUNTO # (1-";EP-SP+1;") ";: INPUT C:
```

(segue)

B sarebbe 8, quello finale 11, la prima linea sarebbe 12 e quella finale 15.

Nel programma le linee 5 - 86 inizializzano la memoria e sottopongono all'utilizzatore le opzioni principali. LOMEM è posto a 16384, così che le variabili non interferiscano con la pagina grafica ad alta risoluzione. Nelle matrici delle dimensioni, T, X e Y sono usate come variabili temporanee e TR, la sola matrice non ancora menzionata, conterrà i valori delle coordinate tridimensionali traslate in due dimensioni.

Tutte le subroutine che riguardano la creazione, la modifica, il caricamento e il salvataggio delle figure vengono considerate routine di servizio, separate dal corpo principale del programma che permette di vedere e manipolare le figure. Queste routine sono nelle righe da 170 a 2020. La routine che crea nuove figure è posta da 1000 a 2020. L'utente batte per primi i punti, poi le linee. Quando ha finito di battere i punti (coordinate X, Y, Z) scrive F per "fine". A questo punto vengono inseriti i numeri degli estremi delle linee, seguiti di nuovo da una F al termine di questa fase del programma.

La subroutine che "edita" le figure è nelle righe da 170 a 239. Permette all'utente di vedere i punti e le linee che compongono le figure e di modificarne i valori a suo piacimento. Mentre state disegnando una figura nuova, può esservi conveniente inserire alcuni punti fittizi a cui non verranno collegate linee, nel caso possano servire in seguito. Nello stesso modo potete inserire linee fittizie (per esempio una linea che connette un punto con se stesso) per un eventuale uso futuro.

Le altre routine in questa sezione di programma servono a salvare una figura (righe 250 - 288), caricarne una (righe 300 - 336), riinizializzare le variabili (riga 350) e salvare l'immagine dello schermo (righe 400 - 420).

Ci sono poi un paio di altre subroutine che seguono quelle elencate, che permettono di scegliere la figura da editare o manipolare, e che compiono i calcoli necessari per la visualizzazione delle figure. Le righe da 2800 a 2875 permettono di inserire il nome di una figura da editare o manipolare. Dalla riga 2900 alla 2930 troviamo la routine che permette di scegliere tra la manipolazione di singole figure o dell'intero insieme di oggetti, assegnando i valori appropriati alle variabili in rapporto alle vostre risposte. NP, NL e NF sono i numeri, rispettivamente, dei punti, delle linee e delle figure. SP e EP sono gli indici iniziale e finale della figura scelta, mentre SL e EL puntano alle linee iniziale e finale.

Segue listato 1.

```
      IF C<1 OR C'EP-SP+1 THEN 6200
6210  C=C+SP-1: FOR I=0 TO 2: CR(I)=P(C,I): NEXT:
      GOTO 6000
6250  POKE-16302,0: GET A$: POKE-16301,0:
      GOTO 6000
6300  INPUT"MOLTIPLICARE PER? ";M: CT=CT*M: C=4:
      RETURN
```

Visualizzazione

Ci sono alcune subroutine dedicate esclusivamente alla procedura di visualizzazione delle figure. Le righe 120 - 165 controllano questa procedura. Per primi devono essere eseguiti i calcoli relativi al centro e alla distanza dell'osservatore. La subroutine 2900 - 3140 svolge questi calcoli, come abbiamo già visto. In un ciclo vengono svolte le seguenti funzioni: vengono calcolate le posizioni dei punti e tradotte nelle coordinate dello schermo, vengono tracciate le linee, l'utente sceglie l'operazione che desidera venga compiuta, vengono cancellate le linee, quindi il ciclo riprende calcolando i nuovi punti, disegnando le linee, ecc. Ogni processo viene svolto da una particolare subroutine. I punti vengono calcolati e trasposti nelle righe 4000 - 4400. Il ciclo in questa subroutine parte dal primo punto della figura e finisce con l'ultimo, svolge l'operazione scelta (C contiene il codice dell'operazione), poi traduce il punto nelle sue coordinate bidimensionali e le memorizza nella matrice TR. Dopo aver elaborato ciascun punto, la subroutine ritorna al programma principale.

Le righe da 5000 a 5290 tracciano o cancellano le righe sul video. SW e FS sono dei commutatori logici che stabiliscono quale compito debba essere svolto. Se SW è uguale a zero, la subroutine cancella le linee una alla volta, mentre se FS è zero, viene cancellato tutto lo schermo e vengono tracciate le nuove linee. Se sia FS che SW sono uguali ad uno, allora vengono tracciate solo le nuove linee (FS=1 se viene modificata una sola figura; in questo modo non vengono cancellate le altre figure durante il movimento). Anche questa subroutine compie un ciclo tra la prima e l'ultima riga, determinando gli estremi tramite la matrice TR, poi controlla che la linea stia entro i limiti dello schermo. La sezione da 5070 a 5270 controlla che ogni estremo stia dentro allo schermo e cerca, se possibile, un segmento che vi sia contenuto. Questo evita i guai provocati dalle eventuali parti della figura che superano i bordi, sopra, sotto o di lato rispetto allo schermo.

L'ultima subroutine, dove vengono prese tutte le decisioni, è posta nelle righe 6000 - 6300. È qui che viene mostrata la lista delle operazioni e vengono definite le costanti. Per facilitare l'uso del programma vediamo ora una lista delle possibili scelte:

1. *Ruotare* Permette la rotazio-

ne della figura. Continuate fornendo direzione e angolo.

2. *Spostare* Muove una figura. Di nuovo dovete inserire la direzione e il numero di unità di cui la figura deve essere mossa.

3. *Scalare* Cambia la scala della figura. Continuate inserendo la costante per la quale le dimensioni della figura devono essere moltiplicate. La costante può essere un numero intero o decimale.

4. *Distorcere* Cambia la scala di una sola dimensione. Scegliete la dimensione (larghezza, altezza o profondità) e la costante moltiplicativa.

5. *Muovere tutto/solo una figura* Permette di scegliere se le operazioni che seguono devono modificare tutto lo schermo o una sola figura. Scegliete poi quale figura intendete modificare.

6. *Centro* Permette di scegliere il centro attorno al quale deve ruotàre la figura. Il centro viene utilizzato anche per cambiare la scala delle figure. Talvolta è conveniente mantenere uno specifico punto per le operazioni di rotazione e cambio scala successive. Con questo comando inserite le coordinate del nuovo centro.

7. *Menù* Riporta alla routine principale di scelta.

8. *Schermo intero* Permette di osservare l'intera figura sullo schermo fino a quando non viene premuto un tasto.

9. *Scala dello schermo* Permette di cambiare la scala di ciò che appare sullo schermo, senza cambiare il valore delle coordinate in memoria. È come osservare gli oggetti con un cannocchiale anziché aumentarne le dimensioni. È utile anche per modificare l'effetto della prospettiva; accade come quando si osserva un oggetto da vicino (maggiore prospettiva apparente) oppure da lontano (minore effetto prospettico). Per avere più prospettiva, spostate l'oggetto molto vicino, e diminuite la scala dello schermo. Per diminuirla, allontanate l'oggetto e ingranditelo con questo comando.

Questo programma dà una buona idea di come i computer simulano la grafica tridimensionale, di quali operazioni si possono compiere e di come ciò sia possibile. ■

Che rischi corre il vostro software?

Le tecniche di protezione del software sono quasi sempre inefficaci. Solo con progetti diversi dei microprocessori si potranno creare dei dispositivi più sicuri

di *J. Commander*

I "pirati" di software, che sfruttano i segreti del linguaggio macchina e si servono dei disassemblatori e dei copiatori bit a bit, gli strumenti più efficaci del loro mestiere, rappresentano il tallone d'Achille degli imprenditori economici nel campo della programmazione dei computer. Le necessità di mercato spingono i programmatori a compiere ogni sforzo per difendere le proprie preziose creazioni, ed i pirati sembrano accettare di buon grado la sfida a chi scopre per primo i segreti di qualche sistema per la protezione del software. Sembra che il pirata di software e molti programmatori stiano vivendo, in questa gara, una nuova affascinante avventura.

I programmatori di microcomputer, che sono abbastanza astuti, e talvolta addirittura geniali, hanno trovato centinaia di accorgimenti per proteggere i loro programmi. Ce n'è di tutti i tipi, semplici, sofisticati, imprevedibili e anche completi. Molti programmatori utilizzano l'hardware o si servono di alcune peculiarità della loro macchina. Ma alla fine tutti questi sforzi si rivelano inutili.

Indipendentemente da come ci si pone di fronte al problema della protezione del software, può essere interessante studiare alcune delle tecniche più usate. Questa panoramica sui trucchi per la protezione dalle copie non ha la pretesa di riportare tutte le idee in ordine cronologico né vuol suggerire il metodo per rendere vana questa o quella tecnica di protezione. Si propone, invece, di mostrare come la protezione del software per microcomputer non possa mai essere assolutamente sicura. Per raggiungere questo traguardo, ambito o meno a seconda del punto di vista, bisognerà attendere una nuova generazione di microprocessori.

Perché proteggere il software?

Un accorgimento per proteggere il software è semplicemente un metodo per rendere impossibile la riproduzione di vari prodotti. Molti programmatori e commercianti di software si servono di queste tecniche per prevenire la duplicazione abusiva di prodotti che hanno richiesto investimenti considerevoli o ingente mole di lavoro.

Un software irriproducibile è il sogno dei commercianti ma molto spesso un incubo per gli utenti. Questi infatti desiderano poter disporre di più copie di un certo prodotto, perché il software per sua natura si presta ad essere distrutto accidentalmente. Questo è il dilemma della protezione del software.

Le vie per risolverlo ci sono. I commercianti possono fornire più copie allo stesso prezzo, sostituire i

dischi danneggiati a prezzo di costo o realizzare metodi di protezione che consentano la copiatura in numero finito. Per l'utente queste soluzioni sono soddisfacenti.

Il grosso guaio dei commercianti, comunque, è che il software non si può proteggere completamente. Inoltre, i trucchi antiriproduzione, oltre ad essere solo uno "scherzetto" per i professionisti, daranno fastidio ad alcuni utenti legittimi. In questo senso i dispositivi di protezione del software sono come piume di pietra sulle ali dei commercianti.

Accorgimenti per la protezione del software

Gli accorgimenti per proteggere il software sono grossomodo classificabili in quattro categorie: quelli che dipendono dall'hardware, quelli per alterazione della procedura di caricamento, software che verifica le condizioni in fase d'esecuzione e quello che viene eseguito tramite un "filtro". Tutti questi generi di accorgimenti richiedono un'interazione tra software e hardware e questo è in definitiva il loro punto debole.

Ogni volta che si usa il software, viene chiamato in causa il microprocessore. I microprocessori, invenzioni meravigliose, sono semplici e fedeli esecutori di istruzioni. Queste ultime possono però essere intercettate in molti modi.

Le tecniche di modifica del formato in fase di caricamento sono il metodo di protezione più comune. Esse funzionano sia su nastri magnetici che su floppy disk. Si basano su un principio concettualmente molto semplice. Molti microcomputer caricano dati dal supporto magnetico utilizzando routine in linguaggio macchina contenute nella ROM (*Read Only Memory*). Ci si può servire di queste routine per caricare da nastro o disco uno speciale caricatore non standard, che a sua volta verrà usato per caricare un programma.

Lo stesso programma sarà scritto in modo da poter essere caricato solo tramite il formato non standard. Ad esempio, molte routine di caricamento da nastro effettuano somme di controllo semplicemente sommando i valori dei byte (caratteri) di un record quando vengono caricati in memoria e le confrontano con il valore somma contenuto nell'ultimo byte del record. Se i due numeri sono uguali, il caricamento continua con il record seguente; altrimenti viene segnalato un errore. Modificando la lettura della somma di controllo si può "ingannare" il microprocessore rendendo il nastro copiabile ma non caricabile.

L'alterazione del formato dei floppy disk può consistere semplicemente nell'aggiungere una pista (per esempio la 41esima su un dischetto a 40 piste) o nel lasciarne una senza formato all'interno del disco. La prima informazione che verrà caricata da disco sarà un caricatore speciale; poi verrà caricato il programma con la routine di caricamento non standard, che si accerterà delle speciali condizioni, come per esempio la presenza della 41esima pista. Un disco riprodotto non avrà questa pista e quindi non potrà essere caricato.

Tutte le tecniche di alterazione del formato possono essere escluse

Si può ripetere questo processo a più livelli; si può programmare un piano di "caricamento nidificato" in cui si carica un caricatore speciale da disco, che a sua volta carica un altro caricatore speciale, che carica il programma. Tuttavia tutte queste tecniche di alterazione del formato di caricamento possono essere eluse, perché la prima informazione caricata contiene il codice delle routine di caricamento non standard. Questo codice può essere disassemblato ed analizzato, ed il pirata può seguire il metodo prevedendo le modifiche al formato.

La protezione hardware contempla di solito il collegamento di un dispositivo ad una porta indirizzabile del calcolatore. Il software viene poi sviluppato in modo tale che per prima cosa il programma controlla se alla porta è collegato il dispositivo giusto: se non è così, l'accesso al programma è precluso. Anche in questo caso, però, la dipendenza dal software è insita nel procedimento, che quindi può essere reso vano.

Molte altre tecniche di protezione sfruttano le peculiarità specifiche dell'hardware dei vari tipi di computer. Con gli Apple, per esempio, si possono scrivere dati sui confini tra le piste dei floppy disk, un buon nascondiglio per byte o routine chiave, senza i quali i programmi non possono essere caricati.

I computer Atari consentono di predisporre un byte normalmente usato per l'operazione di reset a caldo per effettuare un reset a freddo che azzera la memoria. Il software protetto con questo dispositivo è l'unica informazione che può essere contenuta in memoria; qualsiasi tentativo di caricare un monitor prima o dopo il programma, azzera la memoria. Questo dispositivo è connaturato all'hardware dell'Atari ed è perciò sicurissimo.

I videogiochi dell'Atari si basano su hardware progettato appositamente per impedire la duplicazione abusiva del software. Grazie a componenti segreti e poco conosciuti, si è modificato l'hardware perché possa eseguire delle istruzioni in maniera non tradizionale. L'insieme delle istruzioni può essere codificato in modo che una chiamata provochi un salto o una operazione di caricamento di registri; solo l'Atari lo sa di sicuro. Avendo a disposizione abbastanza tempo, denaro e un computer per la codifica, la tecnica può essere decifrata, ma difficilmente vale la pena di farlo.

Gli altri metodi per la protezione di software più noti richiedono un elaboratore molto grande. È necessario che questo verifichi le condizioni in cui il programma viene eseguito; perciò l'esecuzione risul-

terà notevolmente più lenta. Accorgimenti di questo tipo possono servirsi degli *interrupt* di un elaboratore per impedire periodicamente al flusso di programma di indirizzarsi verso una locazione di memoria sconosciuta, verificare un byte e poi tornare all'istruzione seguente l'interrupt.

I programmatori in Basic possono usare alcuni semplici ma efficaci metodi di protezione. Queste tecniche di solito prevedono la disattivazione delle routine di *break* e di *reset*, in modo che, una volta caricato il programma, sia negato l'accesso a livello del sistema. Inoltre i programmi Basic possono contenere dei tab o dei codici di controllo per rendere impossibile il listaggio di un programma.

Molti sistemi operativi su disco offrono anche livelli di protezione a cui si può ricorrere per prevenire l'accesso a un programma in Basic o in linguaggio macchina attraverso il sistema operativo. Tuttavia, vi si può arrivare accedendo alla memoria per l'utente con un monitor per il linguaggio macchina.

Un altro accorgimento di protezione a disposizione dei programmatori di Basic più raffinati consiste nel conservare i programmi Basic nel formato di sistema con variabili predefinite. Questo metodo prevede l'inserimento nel codice di una variabile chiave, memorizzata con una utility speciale, che non deve apparire nel listato del programma. Questa variabile può essere usata in una routine fondamentale che altrimenti provoca la fine del programma. Quindi qualsiasi copia ottenuta usando la routine di I/O del Basic non funzionerebbe.

Proteggere o non proteggere

È probabile che le tecniche di protezione rivestano una certa importanza per parecchio tempo, perché i venditori di software ed i programmatori hanno sempre interesse che le duplicazioni non autorizzate siano minime. Gli accorgimenti di protezione possono essere utili perché molti acquirenti non sono abbastanza esperti tecnicamente da poterli superare. Ma ci saranno pur sempre degli utenti che, provocati nell'orgoglio, supereranno gli ostacoli e distribuiranno delle copie per provare la loro abilità e astuzia.

Ma non c'è proprio un metodo sicuro per proteggere il software? I grossi calcolatori per molti anni hanno contenuto elaboratori particolarmente adatti alla protezione del software. Perché le case costruttrici di microcomputer non possono raggiungere gli stessi risultati?

La situazione storica determinatasi durante l'evoluzione del moderno semiconduttore insegna che il fattore determinante era lo spazio sul chip. I progettisti si sono indirizzati verso lo sviluppo di elaboratori che funzionassero, benché realizzati su un singolo chip; inoltre il progetto doveva consentire la produzione di massa. Di conseguenza i primi microprocessori a 4 bit vennero realizzati con una circuitazione minima, tale da poterne permettere il funzionamento.

Quando nacquero gli elaboratori a 8 bit, venne aggiunto solo il minimo essenziale di circuitazione. Nessuno di preoccupò di pensare a circuiti che potessero essere usati per la protezione del software; erano ritenuti "fronzoli" da ignorare.

Oggi, guardando indietro, vediamo che chi ha progettato i primi elaboratori ha peccato di eccessivo rigorismo; chi li ha seguiti ha continuato sulla stessa strada. Tutto ciò che serve per aggiungere ai microprocessori una protezione software dipendente dall'hardware è una ulteriore coppia di registri e dei circuiti per produrre una relazione di dipendenza tra il software di sistema e quello applicativo.

Nei grossi calcolatori, questa relazione di dipendenza funziona così. La memoria è suddivisa in sezioni a cui ci si può indirizzare in un unico modo. Tale sezionamento è effettuato dall'elaboratore centrale secondo la disponibilità di memoria. L'elaboratore si serve di due registri speciali per memorizzare gli indirizzi degli estremi di una partizione di memoria al momento di utilizzarla. Alla memoria così suddivisa può accedere solo il sistema operativo del computer, e solo quando quest'ultimo si serve del software di sistema. Per raggiungere il codice in qualsiasi partizione di memoria, finché la macchina è in questo stato, si possono usare i monitor o i disassemblatori.

Tuttavia ad un certo punto l'operatore fa passare il sistema al software applicativo. Quando il passaggio è fatto, ogni partizione di memoria si chiude in se stessa. Il controllo non può più passare da una partizione di memoria ad un'altra. Se un programma tenta di raggiungere delle locazioni di memoria esterne alla sua partizione, verrà prodotto un messaggio d'errore e, in certi casi, tale programma viene eliminato dalla memoria.

Per utilizzare questo tipo di sistema master-slave, il costruttore del microcomputer (e i pochi privilegiati che pagheranno per poterlo fare) dovrebbero essere gli unici utenti a cui è consentito di accedere al sistema operativo col software di sistema.

La maggior parte degli utenti sarebbero costretti a comperare il computer con un sistema operativo orientato al software esterno. Inoltre questi potrebbero far funzionare programmi ma non copiarli, a meno che non contengano una routine di copia. Certi programmi di utilità del sistema operativo potrebbero avere limitato accesso nelle partizioni di memoria esterne a quella in cui risiedono, grazie ad una concessione speciale del sistema operativo (cioè del costruttore).

Questo tipo di progetto master-slave può essere compreso nel progetto dei moderni microprocessori. Ed in effetti sorprende che l'ultima generazione di elaboratori non sia dotata di questa possibilità. Prima che possa disporne, ci sarà sicuramente qualche pirata agguerrito, pronto ad affondare i suoi colpi nelle viscere della macchina alla ricerca della chiave della stanza che racchiude il tesoro, il frammento di software "protetto". ■

COMMODORE PET/CBM

L'autoprogrammazione: un primo passo verso l'intelligenza artificiale

Ettore Massimo Albani

La domanda che maggiormente lascia perplessi i programmatori di computer è la seguente: "Può il computer, in particolari condizioni, modificare il proprio programma?". Tale domanda si colloca nell'ambito più generale della "intelligenza artificiale", ma anche un personal computer come il PET/CBM può dare un contributo alla comprensione di tale problema. La routine qui proposta consente di inserire una linea di programma da tastiera mediante un'istruzione INPUT, e cioè mentre il computer sta eseguendo il suo stesso programma!

Nulla vieta di ampliare la routine facendo "calcolare" la linea dallo stesso programma; ad esempio, si potrebbe decidere come e quali variabili del programma visualizzare in certe condizioni.

Presentiamo anche un'applicazione, che potrà essere usata come subroutine in molti programmi gestionali e scientifici: l'INPUT numerico calcolato. Tuttavia, prima di addentrarci nella descrizione della routine, si rende necessaria un po' di teoria.

Come il CBM memorizza un testo Basic

Come è noto, il CBM colloquia con l'operatore in linguaggio Basic; tuttavia, i programmi vengono "tradotti" dall'interprete Basic nel linguaggio macchina usato dal microprocessore 6502. Per la precisione, le linee di programma (contraddistinte dalla presenza del numero di linea) vengono memorizzate nell'area RAM in maniera diversa da come sono scritte dall'operatore, mentre con il comando di LIST vengono ritradotte in formato Basic.

Il CBM, infatti, traduce i comandi Basic mediante un codice esadecimale di un byte: in questo modo qualsiasi comando, per quanto lungo, occuperà sempre un byte di RAM, aumentando, così, la disponibilità di memoria.

La memoria RAM del CBM è divisa in pagine e ogni pagina corrisponde a 256 (o $FF in notazione esadecimale) byte. Le prime quattro pagine, dalla locazione (esadecimale) $0000 alla $03FF, sono riservate al sistema operativo che le usa per memorizzare diverse informazioni, come ad esempio i puntatori delle variabili, quelli di inizio del testo Basic, i buffer delle cassette, ecc. In particolare i primi 256 byte (fino a $FF), appartengono alla cosiddetta "pagina zero". L'area di RAM riservata alla memorizzazione del testo Basic inizia alla locazione $0400 (1024 in decimale). Il semplice programma che segue:

```
100 A=0
110 A=A+1 : PRINT A
120 GOTO 110
```

sarà memorizzato nella RAM in questo modo:

```
0400: 00 09 04 64 00 41 B2 30
0408: 00 16 04 6E 00 41 B2 41
0410: AA 31 3A 20 99 20 41 00
0418: 22 04 78 00 89 20 31 31
0420: 30 00 00 00 AA AA AA AA
```

Il primo byte è sempre 00, mentre i successivi due byte (09 04) indicano la locazione di inizio della successiva riga Basic; si noti che devono essere letti al contrario (prima il secondo poi il primo), ottenendo così la locazione $0409. Questo indirizzo è chiamato *link address*. I successivi due byte rappresentano il numero della riga: 64 00 diventa $0064, che è 100 in decimale.

Inizia a questo punto il testo Basic vero e proprio: $41 corrisponde alla lettera "A", $B2 al simbolo "=", $30 al numero "0". C'è poi un $00: è il separatore fra una riga e la successiva. Alla locazione $0409 inizia infatti la seconda riga.

Ci sono i due byte di link, i due del numero di riga (6E 00 = $006E = 110) ed il testo: $41="A", $B2="=", $41="A", $AA="+", $31="1", $3A=":", $20=" ", $99="PRINT", $20=" ", $41="A", e così via.

Così come $99 corrisponde a "PRINT", anche tutte le altre parole chiave (*tokens*) del Basic hanno il loro codice (vedi tabella 1 in fondo alla rubrica).

La routine

Fatte queste premesse, andiamo ad analizzare la routine. Lo scopo è di aggiungere al programma in esecuzione una linea, di volta in volta introdotta mediante l'istruzione INPUT X$.

Per far questo bisogna memorizzare il contenuto di X$ in alcune locazioni del testo Basic, appositamente tenute disponibili, in modo che la linea divenga eseguibile mediante un GOSUB. Il programma è molto

COMMODORE PET/CBM

semplice nella sua forma primitiva, ma si presta a molte modifiche ed ampliamenti. Segue una descrizione delle singole linee della routine.

```
  1 GOTO00030
  2 :                         : RETURN
 30 INPUT X$
 40 C=0: L=LEN(X$): IF L>23 THEN 30
 50 FOR I=1041 TO 1063: C=C+1
 60 IF C>L THEN Y=32: GOTO140
 70 Z$=MID$(X$,C,1): Y=ASC(Z$)
 80 IF Y=43 THEN Y=170: GOTO140: REM "+"
 90 IF Y=45 THEN Y=171: GOTO140: REM "-"
100 IF Y=42 THEN Y=172: GOTO140: REM "*"
110 IF Y=47 THEN Y=173: GOTO140: REM "/"
120 IF Y=94 THEN Y=174: GOTO140: REM "↑"
130 IF Y=61 THEN Y=178
140 POKE I,Y
150 NEXT
160 END
```

Listato 1.

1: Esegue il salto alla prima riga del programma utente. Si noti che il numero di linea del GOTO è fisso di cinque cifre.
2: Sono 23 spazi disponibili per la memorizzazione della stringa. Si noti la presenza del RETURN che consente di richiamare la linea da qualsiasi punto del programma e la presenza dei due punti iniziali che servono a delimitare gli spazi.
30-40: INPUT e controllo della lunghezza.
50: Inizio del ciclo di memorizzazione; le locazioni 1041 e 1063 sono la prima e l'ultima disponibili nella riga 2.
60: Riempie con dei blank lo spazio eventualmente non occupato fino alla locazione 1063.
70: Ricava uno ad uno i caratteri dalla stringa X$ e pone in Y il corrispondente valore ASCII.
80-120: Questa serie di IF trasforma il codice ASCII degli operatori aritmetici nel corrispondente valore *token* del Basic. Si noti che mancano tutti gli operatori logici (AND, OR e NOT) e le funzioni matematiche (SIN, COS, TAN, SQR, ecc.) che richiedono un programma più complesso.
130: Memorizza il codice nelle locazioni disponibili.
140: Chiusura del ciclo.
150: Fine del programma.
Si noti ancora come le prime due linee di programma sono memorizzate nella RAM: la serie di $20 che inizia alla locazione $0411 sono gli spazi disponibili.

```
0400: 00 0C 04 01 00 89 30 30
0408: 30 33 30 00 2C 04 02 00
0410: 3A 20 20 20 20 20 20 20
0418: 20 20 20 20 20 20 20 20
0420: 20 20 20 20 20 20 20 20
0428: 3A 20 8E 00 -- -- -- --
```

Sviluppi e modifiche

Si è solo accennato, precedentemente, al problema di codificare anche gli operatori logici e le funzioni trigonometriche. L'inconveniente risiede nel fatto che questi operatori occupano, all'interno della stringa, più di un byte, e quindi la loro decodifica risulta più complessa. La soluzione più semplice è quella di

```
    1 GOTO00100
    2 X=                        : RETURN
  100 REM * INIZIO PROGRAMMA UTENTE
  110 .............
      .............
      .............
 1220 INPUT X$: IF LEFT$(X$,1)<>"="
      THEN 1260
 1230 X$=MID$(X$,2): GOSUB 10000
 1240 IF F THEN 1220
 1250 GOTO 1270
 1260 X=VAL(X$)
 1270 .............
      .............
      .............
10000 REM * INPUT CALCOLATO (IN X$, OUT X)
10010 F=0: C=0: L=LEN(X$):
      IF L<23 THEN 10020
10015 PRINT "ESPRESSIONE TROPPO LUNGA":
      F=1: GOTO 10199
10020 FOR I=1042 TO 1063: C=C+1
10030 IF C>L THEN Y=32: GOTO10110
10040 Z$=MID$(X$,C,1): Y=ASC(Z$)
10050 IF Y=43 THEN Y=170: GOTO10110:
                          REM "+"
10070 IF Y=45 THEN Y=171: GOTO10110:
                          REM "-"
10080 IF Y=42 THEN Y=172: GOTO10110:
                          REM "*"
10090 IF Y=47 THEN Y=173: GOTO10110:
                          REM "/"
10100 IF Y=94 THEN Y=174: REM "↑"
10110 POKE I,Y
10120 NEXT
10130 GOSUB 2
10199 RETURN
```

Listato 2.

COMMODORE PET/CBM

codificarli con dei caratteri particolari (ad esempio @ = AND) ed eseguire il riconoscimento come nelle linee 80-120.

Un'altra modifica può essere quella di aumentare lo spazio disponibile nella linea 2, oppure di aggiungere altre linee di blank. Si ricordi che la massima lunghezza di una linea è di 80 caratteri e che i blank devono essere delimitati dai due punti.

Si raccomanda, comunque, di porre molta attenzione a non superare lo spazio disponibile di blank, poiché un solo carattere fuori posto può compromettere la corretta memorizzazione del programma.

Quale esempio pratico di utilizzo della routine, risolviamo un problema che si presenta spesso agli utenti del CBM: l'INPUT numerico calcolato. È infatti impossibile, pena una segnalazione di errore, rispondere "1+2*10/(4+3)" alla richiesta di un INPUT X.

1: Salto alla prima linea del programma utente.
2: In questa linea c'è uno spazio in meno, poiché è stato sostituito il ":" con "X="; in questo modo la variabile in uscita sarà sempre X.
1220: INPUT della variabile stringa X$. Se il carattere iniziale è "=", l'INPUT è da calcolare.
1230: Toglie il carattere "=" da X$ e rimanda alla subroutine. Si noti il particolare uso della funzione MID$ che consente di ricavare la parte a destra della stringa senza conoscerne la lunghezza.
1240: Se F< >0 (errore) ripete l'INPUT.
1250: Salta la linea successiva, che deve essere eseguita solo se l'INPUT è un normale numero.
1260: Assegna ad X il valore di X$.
10010: Inizializza il flag di errore (F) e controlla la lunghezza della stringa.
10015: Scrive il messaggio di errore, pone a 1 il flag e conclude la subroutine.
10020: Inizia il ciclo di decodifica e memorizzazione. Si noti che la locazione iniziale è 1042 e non 1041; infatti alla linea 2 "X=" occupa un carattere in più di ":".
10130: Esegue il calcolo. ■

ESA	DEC.	BASIC	ESA	DEC.	BASIC
80	128	END	B0	176	OR
81	129	FOR	B1	177	>
82	130	NEXT	B2	178	=
83	131	DATA	B3	179	<
84	132	INPUT#	B4	180	SGN
85	133	INPUT	B5	181	INT
86	134	DIM	B6	182	ABS
87	135	READ	B7	183	USR
88	136	LET	B8	184	FRE
89	137	GOTO	B9	185	POS
8A	138	RUN	BA	186	SQR
8B	139	IF	BB	187	RND
8C	140	RESTORE	BC	188	LOG
8D	141	GOSUB	BD	189	EXP
8E	142	RETURN	BE	190	COS
8F	143	REM	BF	191	SIN
90	144	STOP	C0	192	TAN
91	145	ON	C1	193	ATN
92	146	WAIT	C2	194	PEEK
93	147	LOAD	C3	195	LEN
94	148	SAVE	C4	196	STR$
95	149	VERIFY	C5	197	VAL
96	150	DEF	C6	198	ASC
97	151	POKE	C7	199	CHR$
98	152	PRINT#	C8	200	LEFT$
99	153	PRINT	C9	201	RIGHT$
9A	154	CONT	CA	202	MID$
9B	155	LIST	CB	203	GO
9C	156	CLR	CC	204	* CONCAT
9D	157	CMD	CD	205	* DOPEN
9E	158	SYS	CE	206	* DCLOSE
9F	159	OPEN	CF	207	* RECORD
A0	160	CLOSE	D0	208	* HEADER
A1	161	GET	D1	209	* COLLECT
A2	162	NEW	D2	210	* BACKUP
A3	163	TAB(	D3	211	* COPY
A4	164	TO	D4	212	* APPEND
A5	165	FN	D5	213	* DSAVE
A6	166	SPC(	D6	214	* DLOAD
A7	167	THEN	D7	215	* CATALOG
A8	168	NOT	D8	216	* RENAME
A9	169	STEP	D9	217	* SCRATCH
AA	170	+	DA	218	* DIRECTORY
AB	171	-			
AC	172	*			
AD	173	/			
AE	174	↑			
AF	175	AND			

Tabella 1. *Elenco delle parole chiave. Le parole chiave precedute da asterisco sono tipiche del Basic 4.0 Commodore.*

SINCLAIR ZX80/ZX81

Tecniche di velocizzazione

Enrico Ferreguti

Le tecniche di velocizzazione si dividono in due principali gruppi: rendere i cicli ordinati e cercare di essere cauti nella scelta delle funzioni matematiche nei lavori di calcolo. Il prossimo esempio di velocizzazione è un ibrido fra i due.

In alcuni programmi matematici si ripete molte volte una stessa operazione, quindi l'uso di cicli è essenziale.

Se fosse necessario costruire una tavola di J^2/K per i valori di K da 1 a 10 e per valori di K da 1 a 100 potrebbero essere usati due programmi:

```
FOR J=1 TO 10
FOR K=1 TO 100
LET A=J**2/SQR K
NEXT K
NEXT J
```

Tempo di elaborazione 3'48" (modo *fast*)

```
FOR J=1 TO 10
LET B=J**2
FOR K=1 TO 100
LET A=B/SQR K
NEXT K
NEXT J
```

Tempo di elaborazione 2'1" (modo *fast*)

Nella prima versione il quadrato di J viene calcolato 1000 volte ma questo breve calcolo è eseguito nella seconda versione solo 10 volte con evidente guadagno di tempo.

Questi due programmi mostrano quanto sia importante la corretta strutturazione dei cicli, ma anche quanto lento sia lo ZX81 ad eseguire quadrati e radici quadrate.

Non c'è nessuna alternativa all'uso della funzione SQR ma ci sono diverse vie per calcolare quadrati ed altre potenze di numeri.

Per esempio il quadrato di 2 può essere calcolato nei seguenti modi:

```
2+2=4
2*2=4
2**2=4
```

È interessante notare quanto velocemente la macchina trovi i quadrati con questi tre metodi:

```
FOR J=1 TO 1000
LET A=2**2
NEXT J
PRINT "FATTO"
```

Tempo di elaborazione 1'56"

```
FOR J=1 TO 1000
LET A=2*2
NEXT J
PRINT "FATTO"
```

Tempo di elaborazione 8"

```
FOR J=1 TO 1000
LET A=2+2
NEXT J
PRINT "FATTO"
```

Tempo di elaborazione 7"

Non abbiamo molto interesse a cambiare moltiplicazioni con più addizioni, ma si può risparmiare tempo di elaborazione sostituendo le elevazioni a potenza con delle moltiplicazioni multiple.

Possiamo velocizzare delle procedure anche usando, al posto di cifre espresse direttamente, delle variabili. Guardiamo questo esempio:

```
FOR J=1 TO 1000
LET A=10
NEXT J
PRINT "FATTO"
```

Tempo di elaborazione 1'7" (modo *slow*)

```
LET B=10
FOR J=1 TO 1000
NEXT J
PRINT "FATTO"
```

Tempo di elaborazione 1'0" (modo *slow*)

Questa tecnica consuma memoria e variabili, però fa guadagnare qualche secondo nell'elaborazione (guadagno che si fa più consistente in caso di funzioni complicate).

In alcuni computer è conveniente mettere tutte le subroutine all'inizio del programma se le routine sono chiamate ripetutamente.

Questo succede perché la macchina comincia a cercare la linea specificata nell'istruzione GOSUB dalla linea 1, quindi le routine memorizzate all'inizio del programma sono trovate più velocemente di quelle memorizzate alla fine. Lo ZX81 invece è differente perché lavora in maniera contraria, cioè trova prima una routine memorizzata alla fine del programma e poi quella all'inizio.

È più veloce eseguire GOSUB 5000 mille volte che

SINCLAIR ZX80/81

eseguire GOSUB 5 le stesse volte.

```
1 GOTO 10
5 RETURN
10 FOR J=1 TO 1000
15 GOSUB 5
20 NEXT J
25 PRINT "FATTO"
```

Tempo di elaborazione 27″ (modo *slow*)

```
10 FOR J=1 TO 1000
15 GOSUB 5000
20 NEXT J
25 PRINT "FATTO"
30 STOP
5000 RETURN
```

Tempo di elaborazione 26″ (modo *slow*)

Robot

In questo gioco, che applica le tecniche di movimento discusse nel numero scorso della rubrica, siete impegnati nella difesa del mondo, cercando di distruggere le orde di robot che vogliono arrivare alla bomba H per tirare la leva e farla saltare. Avete in dotazione un'astronave che vola ad altezza variabile e che può lanciare una bomba alla volta. Se premete 5 la bomba va a sinistra, se premete 8 la bomba va a destra (per lanciare un'altra bomba bisogna aspettare che la precedente cada a terra).

Per abbattere il robot dovete colpirlo proprio su uno dei due spigoli della testa, ma per distruggerlo completamente bisogna colpirlo esattamente tante volte quante sono indicate dal numero che avete inserito all'inizio come livello di gioco. Usando un livello basso è più facile frenare i robot prima che arrivino alla bomba ma, alla distruzione di ogni robot, si avrà un bonus basso; ad un livello più elevato la distruzione sarà più difficile ma il bonus più consistente.

Notate alle righe 53, 1010, 1120 e altre l'uso della TAB descritto sul numero 3 di *Personal Software*. ■

```
  10 LET A=3
  15 LET SC=0
  20 LET B=3
  25 LET J=0
  30 LET H=INT (RND*10+2)
  40 LET S=0
  50 LET D=0
  52 PRINT AT 21,0;"
"
  53 PRINT AT 17,30;"";TAB 28;
"";TAB 29;"";TAB 30;""
  60 GOSUB 1000
  65 PRINT AT 0,0;"LIVELLO DI GI
OCO (1-10)"
  67 INPUT L
  68 PRINT AT 0,0;"PUNTI:0
"
  70 IF INKEY$="5" THEN GOTO 200
0
  80 IF INKEY$="8" THEN GOTO 201
0
  90 GOSUB 1100
 100 GOTO 70
 500 PRINT AT H,B-3;"         ";TAB
B-3;"        ";TAB B-3;"
"
 510 LET B=3
 520 LET H=INT (RND*10+2)
 525 GOSUB 1000
 530 RETURN
 800 PRINT AT V-1,O-S;" "
 810 GOSUB 1000
 820 GOTO 70
1000 LET A=A+1
1010 PRINT AT 16,A-2;"  ";TAB A
-2;"  ";TAB A-2;"  ";TAB A-2;
"  ";TAB A-2;"  "
1020 IF A=26 THEN GOTO 4000
1030 RETURN
1100 LET B=B+1
1110 IF B>29 THEN GOSUB 500
1120 PRINT AT H,B-3;"  ";TAB B
-3;"  ";TAB B-3;"
"
1130 RETURN
2000 LET S=-1
2005 GOTO 2020
2010 LET S=1
2020 LET V=H
2030 LET O=B-2
2040 GOSUB 1100
2050 PRINT AT V-1,O-1;"   ";TAB
O;"O"
2060 LET O=O+S
2065 LET V=V+1
2070 IF O<0 OR O>26 THEN GOTO 80
0
2100 IF V=16 AND O=A THEN GOTO 3
000
2110 IF V=21 THEN GOTO 800
2120 GOTO 2040
3000 FOR K=1 TO 10
3010 PRINT AT 16,A-1;"  ";TAB A-
1;"  "
3020 PRINT AT 16,A-1;"   ";TAB A-
1;"
"
3030 NEXT K
3035 PRINT AT 15,A+1;" ";AT 15,A
-1;"
"
3037 LET SC=SC+1
3040 LET D=D+1
3045 PRINT AT 0,0;"PUNTI: ";SC
3050 IF D=L THEN GOTO 3500
3060 GOSUB 1000
3070 GOTO 70
3500 LET J=J+1
3505 LET SC=SC+L
3507 PRINT AT 0,0;"PUNTI: ";SC
3510 PRINT AT 16,A-3;"       ";TAB
A-3;"   ";TAB A-3;"
"
3515 PRINT AT 0,10;"BONUS PUNTI:
";L
3520 FOR K=1 TO 50
3530 NEXT K
3540 PRINT AT 16,A-3;"       ";TAB
A-3;"      ";TAB A-3;"
"
3545 PRINT AT 0,10;"
"
3550 LET D=0
3555 LET A=3
3560 GOTO 70
4000 PRINT AT 16,27;"  ";TAB 27;
"  "
4010 PRINT AT 2,0;"FINE GIOCO"
4020 PRINT AT 5,0;"I ROBOT HANNO
"
4030 PRINT "DISTRUTTO IL MONDO "
4040 PRINT AT 9,0;"VUOI UN ALTRO
MONDO","DA DIFENDERE ?"
4050 INPUT J$
4055 CLS
4060 IF J$(1)="S" THEN RUN
```

VIC 20

Come realizzare un listato bidirezionale

Carlo Saraceno

Ora che sono disponibili i moduli di espansione di memoria, è possibile scrivere programmi più lunghi. Risulta più difficile però editarne il contenuto senza una copia stampata da esaminare. L'editing da schermo richiede molto tempo: con 22 caratteri per linea si possono listare solo quattro o cinque linee alla volta. Un LIST veramente utile dovrebbe potersi fermare e continuare, a piacere. Un LIST ideale dovrebbe anche scorrere all'indietro.

Questo piccolo programma svolge efficacemente questi compiti. La linea 63001 determina l'indirizzo di partenza (SA) per ogni memoria del VIC. La linea 63002 calcola il numero di linee (LN) del vostro programma. La linea 63003 predispone lo schermo per listare la linea, quindi continua il programma. È scritta in bianco in modo che non vediate i comandi e mantiene lo schermo libero da echi parassiti per controllare la linea listata.

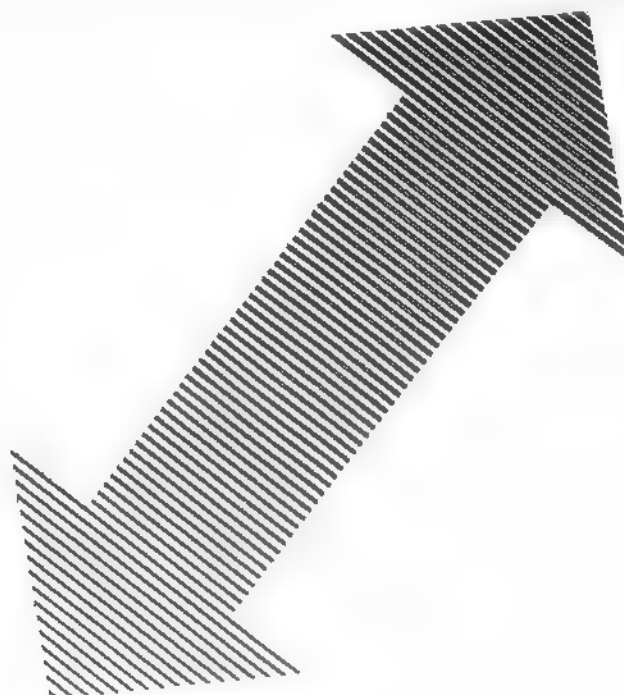
Una volta che una lista è stata iniziata in un programma, il programma termina. Questo perché i comandi del buffer di tastiera nella linea 63004 controllano la lista e quindi continuano il programma con il comando GOTO 63010. Le linee da 63010 a 63030 vi permettono di vedere la linea listata e aspettano che premiate il tasto + per andare avanti alla prossima linea, o – per tornare indietro alla precedente. La linea 63100 controlla il prossimo 0 in Basic, che indica la fine di quella linea Basic, e quindi vi rimanda indietro a calcolare il prossimo numero di linea. La linea 63200 è la routine che controlla la fine della linea precedente. Dovete eliminare le possibilità di trovare uno 0 negli indirizzi che determinano il numero di linea non ammettendo uno 0 in questi due indirizzi. Un altro piccolo trucco vi permette di evitare di battere questo programma dopo aver introdotto un programma principale. Trovate la fine del Basic battendo

CLR: PRINT PEEK(45),: PRINT PEEK(46)

Ora battete la seguente linea che sposta l'inizio del Basic a due byte meno della fine del programma (è necessario o un "nulli" o uno 0 per iniziare a caricare un nuovo programma):

```
63000 REM**  +/- LIST  **
63001 SA=PEEK(44)*256+PEEK(43)-1
63002 LN=PEEK(SA+3)+PEEK(SA+4)*256
63003 PRINT"ᴊ■GOTO 63010":PRINT"LIST";
      LN;
63004 POKE631,19:POKE632,17
63005 POKE633,31:POKE634,13
63006 POKE635,19:POKE636,13
63007 POKE198,6:END
63010 IF PEEK(197)=5 THEN 63100
63015 REM TEST FOR "-" KEY
63020 IF PEEK(197)=61 THEN 63200
63025 REM TEST FOR "+" KEY
63030 GOTO 63010
63100 IF PEEK(SA+5)<>0 THEN SA=SA+1
63105 GOTO 63100
63110 SA=SA+5:GOTO 63002
63200 SA=SA-1:IF PEEK(SA)=0 AND
      PEEK(SA-4)<>0 AND
      PEEK(SA-3)<>0 THEN 63002
63210 GOTO 63200
```

POKE 43,PEEK(45)−2: POKE 44,PEEK(46)

Ora caricate il programma "+/−LIST", resettate i puntatori Basic (POKE 43,1;POKE 44,16 per il VIC senza espansione). Cominciate ad editare battendo RUN 63000. Sarete in grado di verificare il vostro programma linea per linea. Ogni errore scoperto deve essere annotato su carta e corretto in seguito. ■

Gioco del calcio su PET/CBM

di *Carlo Sintini*

Per gli appassionati del calcio, ecco un simpatico gioco abbastanza ben curato nella parte grafica, che gira su qualsiasi PET/CBM (anche con soli 8 K, ma in tal caso occorre eliminare tutte le REM). Chi non avesse problemi di memoria può arricchirlo ulteriormente sonorizzandolo.

Il gioco avviene fra due giocatori (e non contro il PET, che ha quindi solo funzione di arbitro).

Dopo aver dichiarato il nome delle due squadre e fissato il tempo di durata della partita, si devono disporre i giocatori secondo lo schieramento preferito e la partita può iniziare.

Per evitare che i giocatori in difesa attuino un catenaccio impenetrabile concentrandosi in gran numero dentro la propria area di rigore, vale la regola che il portiere è l'unico giocatore autorizzato a rimanere nell'area di rigore.

Ciascuna squadra a turno può scegliere se spostare un suo giocatore o tirare. Il tiro è permesso solo al giocatore in contatto con il pallone (cioè che si trovi in una casella adiacente al pallone, anche in diagonale).

Il tiro può essere forte, medio o debole e in tal caso lo spostamento del pallone sarà rispettivamente di 8, 4 o 2 caselle. Se il pallone incontra un ostacolo (un altro giocatore o il contorno del campo di gioco, eccettuate le porte) rimbalza a caso.

Ad ogni gol lampeggerà per alcuni secondi la scritta GOL nella parte inferiore dello schermo e, quindi, apparirà il punteggio delle due squadre. Nella parte superiore dello schermo invece è sempre visibile il tempo mancante al termine della partita. Dopo ogni gol i giocatori vengono automaticamente risistemati nella formazione iniziale e la palla assegnata alla squadra che ha subito il gol.

La matrice H$ memorizza la posizione dei giocatori e della palla durante la partita mentre la matrice H1$, la formazione iniziale delle due squadre.

Tutti i dati d'ingresso sono trattati con istruzioni GET, in modo che non c'è bisogno di premere dopo ogni comando il tasto RETURN. ■

```
1 REM ********************************************
2 REM *                                          *
3 REM *                  CALCIO                  *
4 REM *             VERSIONE PET CBM             *
5 REM *                                          *
6 REM *            PERSONAL SOFTWARE             *
7 REM *                                          ■
8 REM ********************************************
100 CLR: PRINT CHR$(147)
110 PR=6: PC=10: P$=CHR$(215)
120 DIM H$(11,20), H1$(11,20)
130 PRINTTAB(12) CHR$(18); "GIOCO DEL CALCIO'
140 FOR K=1 TO 40: PRINT CHR$(185);: NEXT
150 FOR K=1 TO 2: Q=3+2*K : GOSUB 3000
160 PRINT "NOME DELLA"; K ; CHR$(157);
170 PRINT CHR$(190): " SQUADRA";: INPUT S$(K)
180 IF LEN(S$(K)) 10 THEN S$(K)=LEFT$(S$(K),8)
190 NEXT
200 Q=11: GOSUB 3000
210 INPUT "QUANTI MINUTI DURERA' LA PARTITA"; MT
220 REM RIEMPIMENTO MATRICE DEL CAMPO
230 FOR K=2 TO 10: FUR J=2 TO 19: H$(K,J)=" "
235 NEXT: NEXT
240 FOR K=2 TO 19: H$(1,K)=CHR$(166)
245 H$(11,K)=CHR$(166): NEXT
250 FOR K=1 TO 11: H$(K,1)="#": H$(K,20)="#"
255 NEXT
260 FOR K=5 TO 7: H$(K,1)=CHR$(255)
265 H$(K,20)=CHR$(255): NEXT
270 REM FINE RIEMPIMENTO
280 GOSUB 540
290 H$(6,10)=P$: Q=11: GOSUB 3000: PRINT TAB(18) P$
300 REM POSIZINAMENTO GIOCATORI
310 CR=211: O$="(A SIN.)": FOR K=1 TO 2
320 Q=23: GOSUB 3000: PRINT CHR$(18); S$(K);
330 PRINT CHR$(146); ":SISTEMA I GIOCATORI ";O$
340 GOSUB 810: GOSUB 780: G$=CHR$(CR): GOSUB 830
350 CR=216: O$="(A DEST.)": NEXT
370 REM MEMORIZZAZIONE POSIZIONE INIZIALE
380 FOR K=1 TO 11: FOR J=1 TO 20: H1$(K,J)=H$(K,J)
385 NEXT J,K
390 REM INIZIO PARTITA
```

(segue)

Segue Gioco del calcio

```
400 TI$="000000": GOSUB 700: T=0
410 T=T+1: IF T>2 THEN T=1
420 Q=23: GOSUB 3000: PRINT S$(T)":" CHR$(18) "M";
425 PRINT CHR$(146) "UOVI O " CHR$(18) "T" CHR$(146);
427 PRINT "IRI"
430 GET QQ$: IF QQ$="" GOTO 430
440 IF QQ$="M" THEN GOSUB 780: GOSUB 1010: GOTO 480
450 IF QQ$="T" THEN GOSUB 780: GOSUB 1460: GOTO 480
460 GOTO 430
470 REM CAMPO DI GIOCO
480 GOSUB 710
490 IF GL=1 THEN GG(T)=GG(T)+1: GL=0: GOSUB 2000
500 IF MM<=0 THEN GOSUB 1910
510 GOTO 410
520 END
530 REM CAMPO DI GIOCO
540 PRINT CHR$(147)
550 FOR K=1 TO 18: PRINTTAB(2) CHR$(K+64) " ";:
555 NEXT
560 FOR K=0 TO 17: FOR J=0 TO 9
565 POKE 32850+80*J+2*K,64: NEXT J,K
570 FOR K=0 TO 18: FOR J=0 TO 8
575 POKE 32889+80*J+2*K,93: NEXT J,K
580 FOR K=0 TO 16: FOR J=0 TO 7
585 POKE 32931+80*J+2*K,91: NEXT J,K
590 FOR K=0 TO 16: POKE 32851+2*K,114
595 POKE 33571+2*K,113: NEXT
600 FOR K=0 TO 7: POKE 32929+80*K,107
605 POKE 32965+80*K,115: NEXT
610 POKE 32849,112: POKE 32885,110
615 POKE 33569,109: POKE 33605,125
620 PRINT: PRINT: FOR K=1 TO 9: PRINT CHR$(K+48);
625 PRINTTAB(38) CHR$(K+48): PRINT: NEXT
630 FOR K=1 TO 18: PRINTTAB(2) CHR$(K+64)" ";: NEXT
640 FOR K=0 TO 16: POKE 32907+40*K,102: NEXT
650 FOR K=0 TO 4: POKE33129+40*K,102
655 POKE 33165+40*K,102: NEXT
660 FOR K=0 TO 3: POKE 33010+K,102
665 POKE 33041+K,102: POKE 33410+K,102
670 POKE33441+K,102: NEXT
680 FOR K=0 TO 8: POKE 33053+40*K,102
690 POKE 33081+40*K,102: NEXT: RETURN
700 REM OROLOGIO
710 MU=VAL(MID$(TI$,3,2))
720 MM=MT-MU: PRINT CHR$(19) CHR$(18);
730 PRINTTAB(3) "MINUTI ALLA SCADENZA DEL TEMPO:" MM
740 IF MM<10 THEN POKE 32804,32
750 MM=MM+1
760 RETURN
770 REM CANCELLAZIONE RIGA
780 FOR Z=0 TO 39: POKE 33688+Z,32: NEXT
790 RETURN
800 REM RITARDO
810 FOR Z=1 TO 3000: NEXT: RETURN
820 REM SISTEM. GIOCATORI
830 FOR N=1 TO 11
840 Q=23: GOSUB 3000: PRINT N CHR$(157) CHR$(190);
845 PRINT " GIOCATORE: COLONNA? ";
850 GET C$: IF C$="" THEN 850
860 PRINT C$: C=ASC(C$): C=C-64
870 IF ASC(G$)=211 AND (C<1 OR C>9) GOTO 840
880 IF ASC(G$)=216 AND (C<1 OR C>18) GOTO 840
890 Q=23: GOSUB 3000: PRINTTAB(28)"RIGA? ";
900 GET R$: IF R$="" GOTO 900
910 PRINT R$: R=ASC(R$): R=R-48
920 IF R<1 OR R>9 GOTO 890
930 IF C=9 AND R=5 THEN N=N-1: GOTO 970
940 R1=R: C1=2*C: R2=R+1: C2=C+1
950 IF H$(R2,C2)<>" " THEN N=N-1: GOTO 970
960 Q=R1*2+1: GOSUB 3000: PRINTTAB(C1) G$
965 H$(R2,C2)=G$
970 GOSUB 780
980 NEXT
990 RETURN
1000 REM MOSSE DEI GIOCATORI
1010 Q=23: GOSUB 3000: PRINT "QUALE GIOCATORE ? ";
1015 PRINT "COLONNA: ";
1020 GET CQ$: IF CQ$="" GOTO 1020
1030 PRINT CQ$: CQ=ASC(CQ$): CQ=CQ-64
1040 IF CQ<1 OR CQ>18 GOTO 1010
1050 Q=23: GOSUB 3000: PRINTTAB(30) "RIGA: ";
1060 GET RQ$: IF RQ$="" GOTO 1060
1070 PRINT RQ$: RQ=ASC(RQ$): RQ=RQ-48
1080 IF RQ<1 OR RQ>9 GOTO 1050
1090 IF T<>1 OR H$(RQ+1,CQ+1)<>CHR$(211) GOTO 1100
1095 GOSUB 780: GOTO 1120
1100 IF T<>2 OR H$(RQ+1,CQ+1)<>CHR$(216) GOTO 1110
1105 GOSUB 780: GOTO 1120
1110 GOSUB 780: GOTO 1010
1120 GOSUB 780: Q=23: GOSUB 3000
1125 PRINT "DIREZIONE? (INTORNO AL 5)";
1130 GET A$: IF A$="" GOTO 1130
1140 A=ASC(A$)-48: IF A=5 GOTO 1130
1150 PRINT "  N. PASSI?"
1160 GET XX$: IF XX$="" GOTO 1160
1170 XX=ASC(XX$)-48: IF XX<1 OR XX>9 GOTO 1160
1180 FOR WW=1 TO XX: FD=0
1200 IF A=1 THEN C=CQ-1: R=RQ+1: GOTO 1290
1210 IF A=2 THEN C=CQ: R=RQ+1: GOTO 1290
1220 IF A=3 THEN C=CQ+1: R=RQ+1: GOTO 1290
1230 IF A=4 THEN C=CQ-1: R=RQ: GOTO 1290
1240 IF A=6 THEN C=CQ+1: R=RQ: GOTO 1290
1250 IF A=7 THEN C=CQ-1: R=RQ-1: GOTO 1290
1260 IF A=8 THEN C=CQ: R=RQ-1: GOTO 1290
1270 IF A=9 THEN C=CQ-1: R=RQ-1: GOTO 1290
1280 IF A=5 OR A<1 OR A>9 THEN GOSUB 780: GOTO 1010
1290 IF H$(R+1,C+1)=" " GOTO 1300
1295 A=INT(RND(1)*9+1): FD=1
1300 IF FD=1 AND A=5 THEN 1290
1310 IF FD=1 THEN FD=0: GOTO 1200
1320 IF T<>1 GOTO 1330
1325 H$(R+1,C+1)=CHR$(211): H$(RQ+1,CQ+1)=" "
1327 MG$=CHR$(211): GOTO 1340
1330 H$(R+1,C+1)=CHR$(216): H$(RQ+1,CQ+1)=" "
1335 MG$=CHR$(216)
1340 FOR Z=1 TO 5
1350 Q=RQ*2+1: GOSUB 3000: PRINTTAB(CQ*2) MG$
1360 Q=RQ*2+1: GOSUB 3000: PRINTTAB(CQ*2) " "
1370 NEXT
1380 FOR Z=1 TO 5
1390 Q=R*2+1: GOSUB 3000: PRINTTAB(C*2) " "
1400 Q=R*2+1: GOSUB 3000: PRINTTAB(C*2) MG$
1410 NEXT
1420 CQ=C: RQ=R
1430 NEXT
1440 GOSUB 780: RETURN
1450 REM TIRI DEI GIOCATORI
1460 Y=0
1470 TG$=CHR$(211+5*(T-1))
1490 Q=23: GOSUB 3000: PRINT"TIRO " CHR$(18) "F";
1493 PRINT CHR$(146) "ORTE, " CHR$(18) "M";
1495 PRINT CHR$(146) "EDIO O " CHR$(18) "D";
1497 PRINT CHR$(146) "EBOLE ?"
1500 GET A$: IF A$="" GOTO 1500
1510 IF A$="F" THEN FT=8: GOTO 1550
1520 IF A$="M" THEN FT=4: GOTO 1550
1530 IF A$="D" THEN FT=2: GOTO 1550
1540 GOTO 1500
1550 FOR K=-1 TO 1: FOR J=-1 TO 1
1560 IF H$(PR+K,PC+J)=TG$ THEN Y=Y+1
1570 NEXT: NEXT
1580 IF Y<>0 GOTO 1600
1585 GOSUB 780: Q=23: GOSUB 3000
1590 PRINT "NON PUOI TIRARE: MUOVI!!": GOSUB 810
1595 GOSUB 780: GOSUB 1000: RETURN
1600 GOSUB 780
1610 Q=23: GOSUB 3000
1615 PRINT "IN QUALE DIREZIONE? (INTORNO AL 5)"
1620 GET A$: IF A$="" GOTO 1620
1630 A=ASC(A$)-48
1640 IF A=5 GOTO 1620
1650 GOSUB 780: FOR XX=1 TO FT
1660 IF A=1 THEN C=PC-1: R=PR+1: GOTO 1740
1670 IF A=2 THEN C=PC: R=PR+1: GOTO 1740
1680 IF A=3 THEN C=PC+1: R=PR+1: GOTO 1740
1690 IF A=4 THEN C=PC-1: R=PR: GOTO 1740
1700 IF A=6 THEN C=PC+1: R=PR: GOTO 1740
1710 IF A=7 THEN C=PC-1: R=PR-1: GOTO 1740
1720 IF A=8 THEN C=PC: R=PR-1: GOTO 1740
1730 IF A=9 THEN C=PC+1: R=PR-1
1740 IF H$(R,C)<>CHR$(255) GOTO 1750
1745 GL=1: A=0: NEXT: RETURN
1750 IF H$(R,C)=" " GOTO1780
1760 A=INT(RND(1)*9+1): IF A=5 GOTO 1750
1770 GOTO 1660
1780 H$(PR,PC)=" ": H$(R,C)=P$
1790 Q=2*PR-1: FOR Z=1 TO 2
1800 GOSUB 3000: PRINTTAB(PC*2-2) P$
```

(segue)

Segue Gioco del calcio

```
1810 GOSUB 3000: PRINTTAB(PC*2-2) " "
1820 NEXT
1830 Q=2*R-1: FOR Z=1 TO 2
1840 GOSUB 3000: PRINTTAB(C*2-2) " "
1850 GOSUB 3000: PRINTTAB(C*2-2) P$
1860 NEXT
1870 PC=C: PR=R
1880 NEXT
1890 RETURN
1900 REM FINE PARTITA
1910 PRINT CHR$(147)
1920 IF GG(1)<>GG(2) GOTO 1927
1923 PRINT "AVETE PAREGGIATO " GG(1) " A " GG(1)
1925 GOTO 1950
1927 Z=(SGN(GG(2)-GG(1))+3)/2
1930 PRINT "HA VINTO " CHR$(18) S$(Z) CHR$(146);
1940 PRINT " PER " GG(Z) " A " GG(3-Z)
1950 Q=4: GOSUB 3000: PRINT "UN'ALTRA PARTITA?"
1960 GET A$: IF A$="" GOTO 1960
1970 IF A$="S" GOTO 100
1980 Q=7: GOSUB 3000: PRINT "OK - CIAO!!": END
1990 REM STAMPA PUNTEGGI
2000 Q=23: FOR B=1 TO 20
2010 GOSUB 3000: PRINTTAB(18) CHR$(18) "GOL"
2020 GOSUB 3000: PRINTTAB(18) "   ": NEXT
2030 GOSUB 3000: PRINT CHR$(18) S$(1) CHR$(146);
2033 PRINT ":PUNTI" GG(1) "    " CHR$(18) S$(2);
2035 PRINT CHR$(146) ":PUNTI" GG(2)
2040 GOSUB 800: GOSUB 780: GOSUB 2080: RETURN
2070 REM SISTEMAZIONE GIOCATORI DOPO IL GOL
2080 FOR K=1 TO 11: FOR J=1 TO 20: H$(K,J)=H1$(K,J)
2085 NEXT: NEXT
2090 IF T=2 THEN PR=6: PC=10: GOTO 2140
2100 H$(6,10)=" "
2110 IF H$(6,11)<>" " GOTO 2120
2115 H$(6,11)=P$: PR=6: PC=11: GOTO 2140
2120 IF H$(5,11)<>" " GOTO 2130
2125 H$(5,11)=P$: PR=5: PC=11: GOTO 2140
2130 IF H$(7,11)<>" " GOTO 2140
2135 H$(7,11)=P$: PR=7: PC=11: GOTO 2140
2140 FOR K=1 TO 9: FOR J=1 TO 18
2150 Q=K*2+1: GOSUB 3000: PRINTTAB(J*2) H$(K+1,J+1)
2160 NEXT: NEXT
2170 RETURN
2999 STOP
3000 PRINT CHR$(19);: FOR HT=1 TO Q
3010 PRINT CHR$(17);: NEXT: RETURN
```

Se convertite questo programma per il vostro computer, spediteci il listato, una paginetta di spiegazioni e un supporto magnetico (disco o cassetta) con la registrazione del programma.

Questo programma è disponibile su disco. Vedete nelle ultime pagine il "Servizio programmi".

PROGRAMMI DI MATEMATICA E STATISTICA

Leggendo questo libro il lettore potrà formarsi quella logica di base indispensabile per la risoluzione di problemi di matematica e statistica.
Ad ogni programma viene preposta un'esposizione schematica del metodo numerico e delle tecniche di programmazione utilizzate, il diagramma a blocchi relativo all'algoritmo, il listato (anch'esso ottenuto da calcolatore) in cui tra l'altro vengono specificati il tempo e la quantità di memoria impiegati.

Cod. 522D **L. 16.000 Pagg. 228**

INTRODUZIONE AL PASCAL

Il volume, incentrato su numerosissimi esempi che verificano costantemente l'apprendimento del lettore, insegna a conoscere, capire ed usare tutte le particolarità e i vantaggi di questo linguaggio. Nel corso della trattazione vengono ampiamente utilizzate le tecniche di programmazione strutturata, come pure tecniche particolari, quali il trattamento dei file, l'utilizzazione della recursività e il trattamento grafico.

Cod. 516A **L. 30.000 Pagg. 484**

COMPUTER GRAFICA

Si può dire che la computer grafica si pone nel contesto più generale del trattamento dell'informazione, avendo individuato nell'immagine un contenuto informativo che è possibile elaborare.
Quest'opera, con il suo rigore informativo e scientifico, si pone come fondamentale nel carente panorama italiano; inoltre le informazioni e gli spunti contenuti nel testo contribuiranno certamente alla divulgazione ed alla formazione di idee nuove e feconde.

Cod. 519P **L. 29.000 Pagg. 174**

APPLE II - Guida all'uso

Se possedete un Apple e volete conoscerlo a fondo, se volete comprarlo, o se semplicemente volete imparare la sua programmazione, troverete in questo libro, tutte le risposte, comprese alcune vere "primizie" che vi occorrono per una perfetta operatività del sistema. Conoscerete i vari componenti del sistema e come usarli al meglio. Verrete guidati alla programmazione in BASIC e a usare le caratteristiche grafiche e sonore del sistema. Imparerete a memorizzare su disco sia programmi che archivi dati, come ad inserire un programma scritto in assembler in uno scritto in BASIC.
E poi ancora, tutte le istruzioni e funzioni BASIC e ben 12 appendici veramente basilari.

Cod. 331P **L. 26.000 Pagg. 400**

CEDOLA DI COMMISSIONE LIBRARIA

Ritagliare (o fotocopiare) e inviare a
Gruppo Editoriale Jackson Via Rosellini, 12 - 20124 Milano

Nome e Cognome ____________________

____

Indirizzo ____________________

____

Cap. ________ Città ____________________ Provincia ________

Partita I.V.A. (indispensabile per le aziende)

☐☐☐☐☐☐☐☐☐☐☐ Si richiede l'emissione della fattura ☐

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

☐ Pagherò al postino il prezzo indicato + L 2 000 per contributo fisso spese di spedizione

☐ Allego assegno n° ____________ di L ____________

Data ____________ Firma ____________

☐ Non Abbonato ☐ Abbonato sconto 10% ☐ L'Elettronica ☐ Elettronica Oggi ☐ Automazione Oggi ☐ Elektor ☐ Informatica Oggi ☐ Computerworld ☐ Bit ☐ Personal Software ☐ Strumenti Musicali ☐ Videogiochi

... dalla libreria JACKSON

Questa rubrica è un'appendice della "Posta dei lettori" dedicata specificamente agli errori riscontrati nei listati che abbiamo pubblicato nei numeri precedenti. Se pensate di aver trovato un "bug" in un programma di *Personal Software*, scrivete a

Personal Software
Rubrica "Debug"
Via Rosellini 12
20124 Milano

Pubblicheremo la vostra lettera con i commenti della redazione, rendendo così un servizio ai lettori. Raccomandiamo soprattutto estrema chiarezza nell'illustrare l'errore che pensate di aver individuato. Non fate come il lettore A.A. di Rovigo che, a proposito del gioco del NIM (*Personal Software* 2, pag. 42), ci scrive:

> Il programma... afferma presuntuosamente che il calcolatore sarà imbattibile se muoverà per primo. Ciò non è assolutamente vero, in quanto, per il fatto che il programma prevede che i numeri dei fiammiferi vengano dati da GET (da INPUT è la stessa cosa), praticamente il computer non può mai vincere. Sono pronto a scommettere.

Il gioco del NIM

Vi informo della presenza di un errore nel listato del "Gioco del Nim" (*Personal Software 2*, pag. 41).

La linea interessata è la 4015 che fa parte della subroutine per la ricerca del numero massimo di fiammiferi e della riga su cui si trovano. In essa l'istruzione W=W+1 genera un errore nella mossa del computer quando questo ne deve eseguire una a caso tra quelle possibili (linee 3090-3100). Ciò accade perché si può uscire dalla subroutine con un valore di W che non corrisponde a quello della riga avente il massimo numero di fiammiferi.

Nell'esempio seguente, si esce con W=1 e M=3 quando invece la riga 1 non contiene più fiammiferi.

Posizione iniziale	1	I I I
	2	I I I
	3	I I I
Il giocatore toglie	1	
3 fiammiferi dalla	2	I I I
prima riga	3	I I I
Muove il computer	1	I I
	2	I I I
	3	I I I

Pertanto l'istruzione W=W+1 va sostituita con W=H.

Riterrei anche opportuno aggiungere la linea

```
322 IF VAL(R$)=0 THEN 320
```

per evitare che il giocatore risponda 0 alla domanda "QUANTI FIAMMIFERI", risposta che attualmente viene accettata anche se errata.

Ing. Ciro Golia
Aversa

Grazie per le sue corrette indicazioni. In effetti il computer oltre a giocar bene a suo favore, addirittura barava, nel senso che talvolta non solo non toglieva nessun fiammifero, ma ne aggiungeva, come nell'esempio che ci ha illustrato. La sua lettera è un esempio, che consigliamo ai lettori di imitare, di come illustrare con chiarezza un errore in un programma. Grazie.

Il gioco del 15

Scrivo per comunicarvi due errori nel listato del programma "Gioco del 15" versione Apple II (*Personal Software* 1, pag. 76). La riga 210 deve leggersi

```
210 IF ASC(M$)=70 THEN END
```

il programma infatti attende il carattere "E" (codice ASCII 69) invece del carattere "F" (codice ASCII 70) per finire.

La riga 270 deve scomporsi in due righe

```
270 IF N(I,J)=M GOTO 280
275 NEXT J,I
```

Infatti il programma alle righe 260 e 270 dovrebbe in teoria ricercare nella tabella il numero che si è impostato e, avendolo trovato, dovrebbe inviare il controllo a riga 280 (e seguenti) per verificare se la mossa è tra quelle permesse, nel qual caso dovrebbe aggiornare la situazione e ricominciare o, in alternativa, dare il MESSAGGIO "MOSSA NON PERMESSA". Ma la riga 270 così come sta include la fine del ciclo nelle istruzioni la cui esecuzione è subordinata al verificarsi della condizione IF e non basta, infatti data la struttura della riga 270, il

NEXT J,I non verrà in ogni caso eseguito. Difatti se al primo tentativo N(1,1)=M allora viene effettuato il salto alla riga 280 e l'esecuzione prosegue senza trovare il NEXT J,I altrimenti se N(1,1) ≠ M allora vengono ignorate le istruzioni che seguono l'IF e l'esecuzione prosegue (comunque) verso riga 280 ancora senza mai trovare il NEXT J,I naturalmente con effetti del tutto indesiderati.

Attilio Grop
Porpetto (Udine)

Le cose stanno esattamente così. Grazie anche a lei per le preziose e corrette puntualizzazioni.

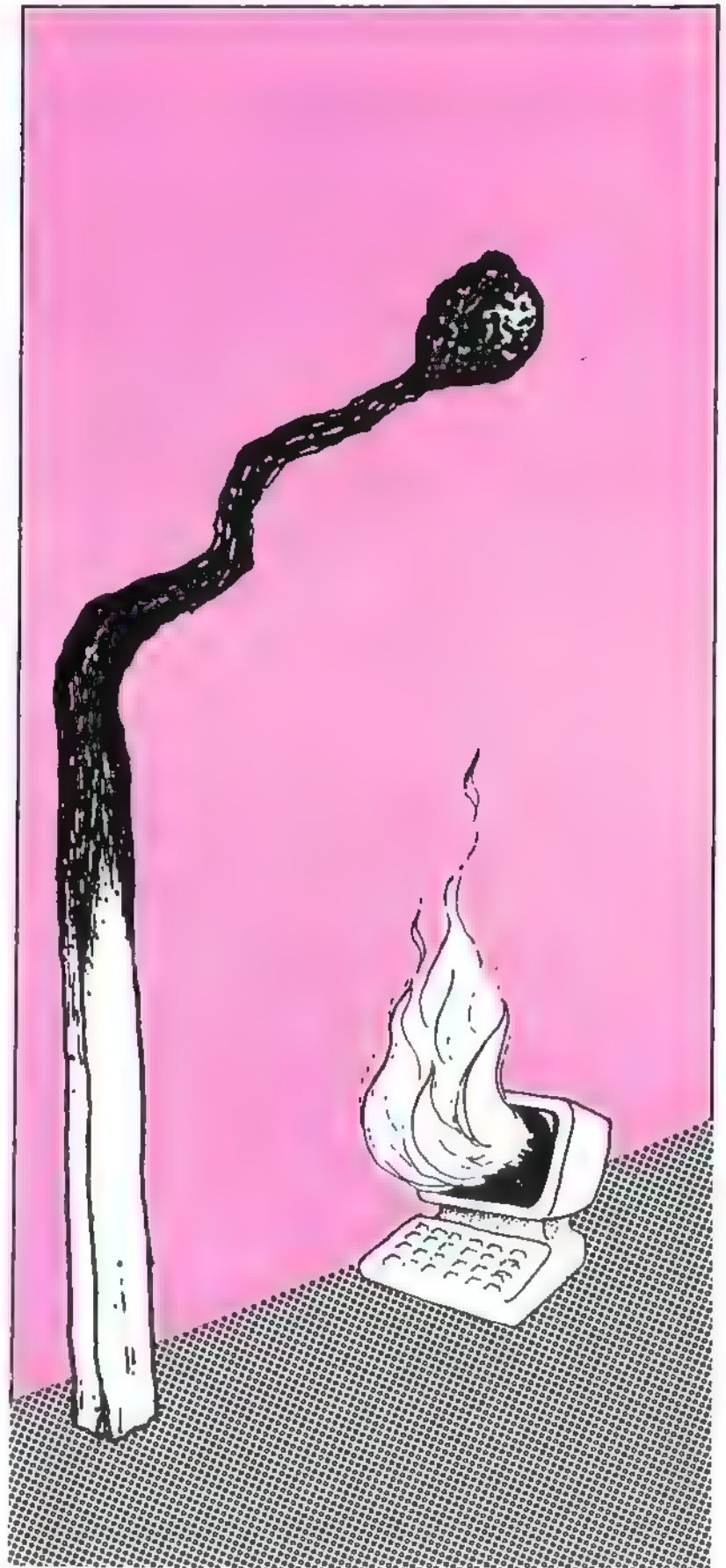
Triste a dirsi, credo di aver trovato un errore (grave) nel programma "Gioco del 15". L'errore è nell'algoritmo di generazione della disposizione iniziale dei 15 tasselli numerati nella matrice 4 × 4; tale algoritmo si basa infatti sulla convinzione (errata) che da qualunque posizione possibile dei tasselli si possa sempre giungere al risultato finale usando soltanto le mosse permesse.

In realtà delle 16! (circa 2×10^{13}) possibili disposizioni iniziali dei tasselli soltanto esattamente la metà di esse sono risolvibili nel senso da noi inteso. Nei rimanenti 10^{13} (circa) casi, il meglio che si potrà ottenere sarà:

1	2	3	4
5	6	7	8
9	10	11	12
13	15	14	—

e non vi sarà alcun modo per sistemare il 14 prima del 15 senza distruggere l'ordine degli altri tasselli (a meno di non scardinare la scatola del gioco). Purtroppo non sono riuscito a ritrovare l'articolo in cui il "maestro" Martin Gardner chiarisce questo aspetto del gioco...

Francesco Balena
Bari

Proprio così. In termini matematici si dice che la permutazione ottenuta è di parità dispari, e siccome quella da ottenere è di parità pari, è impossibile attraversare la barriera: o si è di qua o di là. Un altro lettore, Giancarlo Prunotto di S. Pietro Moncalieri (Torino) che ci ha scritto una lettera analoga ha trovato l'articolo di Gardner: è a pag. 74 del primo volume di Enigmi e giochi matematici *(Sansoni). Proposte per eliminare l'inconveniente: partire dalla disposizione finale ed applicare ad essa alcune centinaia di mosse casuali, in modo da rimescolare i tasselli (Balena), oppure controllare la parità della permutazione contando il numero di scambi per ordinare il vettore (Prunotto).*

Raccogliamo in questa rubrica le conversioni dei programmi che abbiamo pubblicato nei numeri precedenti, realizzate dai lettori per i loro personal. Invitiamo tutti a contribuire, inviandoci un listato, meglio se anche su supporto magnetico, e una descrizione (un paio di pagine dattiloscritte) dei problemi incontrati nella conversione, e delle tecniche usate per risolverli. Inviate la corrispondenza a

Personal Software
Rubrica "Conversioni"
Via Rosellini 12
20124 Milano

Per cominciare pubblichiamo due conversioni del generatore di labirinti pubblicato nel numero scorso (*Personal Software* 3, pag. 67): una è per l'Apple II e una per lo ZX81.

Generatore di labirinti per Apple II e ZX81

Apple II

Gent. Redazione,
appena ho letto sul terzo numero della vostra rivista, il programma per la creazione di labirinti sulla pagina di testo del PET/CBM, ho pensato subito di adattarlo al mio computer.

Come si sa, le locazioni di memoria riguardanti la pagina di testo dell'Apple vanno dalla 1024 alla 2039, e fin qui nulla di eccezionale.

L'inconveniente sta nel fatto che, pokando in ordine successivo tali locazioni, si vedono comparire sullo schermo le righe riempite con il carattere stabilito, ma non progressivamente dalla prima alla ventiquattresima, bensì con dei salti ben precisi: in pratica vengono riempite nell'ordine la prima riga, la nona, la diciassettesima, poi la seconda, la decima, la diciottesima e così via fino all'ottava, sedicesima e ventiquattresima.

Il generatore di labirinti non può ovviamente lavorare in tali condizioni, ma ha bisogno di lavorare su numeri ordinati progressivamente.

Ora proviamo a scrivere sull'Apple il seguente programma:

```
10 HOME: A=1024
20 FOR L=0 TO 80 STEP 40
30 FOR K=0 TO 896 STEP 128
40 FOR I=0 TO 39
50 POKE (A+I+K+L),193
60 NEXT: NEXT: NEXT
```

Adesso diamo il RUN e vedremo, come per incanto, che verranno riempite con delle A tutte le righe, dalla prima alla ventiquattresima, ma questa volta in perfetto ordine progressivo.

A questo punto il problema è risolto. Basta dimensionare un vettore, che chiamiamo S, di (80+960) elementi e creare un algoritmo che ponga S(H), per H da 80 a 1039, uguale di volta in volta alle locazioni da 1024 a 2039 nell'ordine imposto dalla ROM dell'Apple. È necessario che si lascino 80 elementi liberi (da 0 a 79) per evitare che si abbiano valori negativi durante la generazione del labirinto.

In questo modo il programma potrà lavorare sugli elementi del vettore e saremo sicuri che i contrassegni o il carattere voluto verranno piazzati sullo schermo al posto giusto.

Questa operazione indispensabile porta via quasi 14 secondi, ma ovviamente viene effettuata solo al momento del RUN e non è più necessario ripeterla per la creazione dei successivi labirinti.

Comunque, chi ha poca pazienza e possiede il TASC (per compilare il Basic in linguaggio macchina) potrà ridurre l'attesa a soli 3 secondi.

Analisi del programma

Le righe 110-170 servono a fare scrivere la presentazione del programma e a definire le stringhe da visualizzare durante l'esecuzione dello stesso.

La riga 210 dimensiona i due vettori S e Z e pone A = 1024 (prima locazione di memoria della pagina di testo).

La riga 220 contiene l'algoritmo di cui abbiamo parlato, che permette di porre gli elementi del vettore S, da 80 a 1039, uguali alle locazioni da 1024 a 2039 seguendo l'ordine che ci interessa.

La riga 310 serve a riempire con spazi *inverse* il campo entro cui dovrà snodarsi il labirinto e naturalmente si può modificare a volontà seguendo i criteri chiaramente spiegati nel programma per il PET (*Personal Software*, 3 pag. 67).

Le righe 410-460 contengono le istruzioni per creare il labirinto con strisce in positivo (nero) su sfondo inverso (bianco); esse riproducono pratica-

mente il concetto delle istruzioni 200-250 del programma per il PET/CBM.

La riga 510 fa scrivere a pie' di pagina che l'utente deve premere il tasto T per fare comparire il topo (nel nostro caso un asterisco); al posto della solita istruzione PRINT ho usato un'istruzione con cui vengono pokate le locazioni da 2000 a 2039 con il valore del codice ASCII (+128 per avere la stampa in modo normale) dei singoli caratteri della stringa già definita in precedenza.

La riga 520 blocca il programma finché non viene premuto il tasto T.

La riga 610 fa cambiare la didascalia a fondo pagina e fa comparire l'asterisco, che dovrà percorrere avanti e indietro il labirinto all'infinito, o almeno finché vorrà l'utente.

Le righe 1010-1070 contengono le istruzioni che regolano il percorso del topo. Anche questo gruppo di istruzioni rispecchia il concetto applicato nelle istruzioni 1000-1030 del programma per il PET.

In pratica, cosa deve fare il programma?

Esso deve fare partire il topo dalla cella d'inizio del labirinto, con il vettore di direzione puntato in una direzione qualsiasi (nel mio programma ho messo G = 0, ma va bene qualsiasi valore compreso fra 0 e 3).

Prima di fare spostare il topo, dovrà esaminare il contenuto delle locazioni di memoria corrispondenti alle celle adiacenti e questo si ottiene facendo ruotare il vettore di direzione in senso orario ogni volta di 90°, fino a trovare la cella giusta in cui fare spostare il topo. Questo va bene finché il percorso si mantiene rettilineo. Alla prima deviazione, però, bisogna fare in modo che il vettore di direzione, prima di effettuare la rotazione oraria di ricerca, ruoti di 180° cioè assuma il senso inverso a quello che aveva.

Se non si usasse questo accorgimento, il vettore ruoterebbe subito di 90° e, non trovando una cella di labirinto, ruoterebbe ancora di 90°: a questo punto troverebbe certamente una cella di labirinto, ma sarebbe esattamente quella da cui proviene il topo, che così tornerebbe sui suoi passi.

In conclusione il topo partirebbe dalla cella d'inizio, percorrerebbe il primo tratto rettilineo e tornerebbe al punto di partenza, per poi ripercorrere lo stesso tragitto avanti e indietro all'infinito senza mai imboccare la deviazione.

La riga 1040 del mio programma contiene l'istruzione per fare ruotare il vettore di direzione di 180°.

La riga 1070 contiene l'istruzione per farlo ruotare di 90° in senso orario.

```
1040 G=G−2+4*(G<2)
1070 G=G−1+4*(G=0)
```

```
110   A$
      "   PREMI  T  PER FARE COMPARIRE IL TOPO
      ":
      B$=
      "  RET  PER RICOMINCIARE, ESC  PER FINI
      RE"
120   C$=
      "
      ": REM [illegible]
130   D$=
      " GENERAZIONE DEL LABIRINTO [illegible]
      [illegible]"
140   HOME: HTAB [illegible]: PRINT
      "GENERATORE  DI  LABIRINTI"
190   VTAB 2[illegible]: HTAB 1[illegible]: PRINT
      "ATTENDI, PREGO"
210   DIM S([illegible]),Z([illegible]): A=[illegible]
      REM LOCAZIONE INIZIO RAG. TESTO
220   FOR [illegible] TO 8[illegible] STEP [illegible]:
      FOR [illegible] TO 9[illegible] STEP 128: FOR I=0 TO [illegible]
      S(H+8[illegible])=[illegible]+I+[illegible]: H=H+1: NEXT: NEXT:
      NEXT
300   REM
310   HOME: FOR O=2000 TO 2039:
      POKE O,ASC(MID$(D$,O-1999,1))+128:
      NEXT
320   FOR W=1 TO 23: VTAB W: HTAB 2: INVERSE:
      PRINT C$;: NEXT: NORMAL
400   REM
410   WL=32: HL=160: Z(0)=2: Z(1)=-80:
      Z(2)=-2: Z(3)=80: J=122: POKE S(J),180
420   G=INT(4*RND(1)): X=G
430   IF PEEK(S(J+Z(G)))=WL THEN POKE S(J+
      Z(G)),G+176: POKE S(J+Z(G)/2),HL:
      J=J+Z(G): GOTO 420
440   G=(G+1)*(G<3):
      REM ROTAZIONE ANTIORARIA DI 90 GRADI
450   IF G<>X THEN 430
460   G=PEEK(S(J))-176: POKE S(J),HL:
      IF G<4 THEN J=J-Z(G): GOTO 420
500   REM
510   FOR O=2000 TO 2039:
      POKE O,ASC(MID$(A$,(O-1999),1))+128:
      NEXT
520   VTAB 24: HTAB 11: GET K$:
      IF K$<>"T" THEN 520
600   REM
610   FOR O=2000 TO 2039:
      POKE O,ASC(MID$(B$,(O-1999),1))+128:
      NEXT
1000  REM
1010  J=122: POKE S(J),170: G=0
1020  N=J+Z(G)/2:
      IF PEEK(S(N))=HL THEN POKE S(N),170:
      POKE S(J),HL: J=N: GOTO 1040
1030  GOTO 1050
1040  G=G-2+4*(G<2): REM ROTAZIONE 180 GRADI
1050  IF PEEK(-16384)=13 THEN 300:
      REM SE RET RICOMINCIA
1060  IF PEEK(-16384)=27 THEN HOME: END:
      REM SE ESC FINE DEL PROGRAMMA
1070  G=G-1+4*(G=0):
      REM ROTAZIONE ORARIA DI 90 GRADI
1080  POKE-16368,0: GOTO 1020
```

```
520   VTAB 24: HTAB 11: GET K$:
      IF K$<>"T" THEN 520
600   FOR Q=2001 TO 2079:
      POKE Q,ASC(MID$(C$,Q-2000,1))+128:
      NEXT
1002  FOR Q=200 TO 120 STEP-40:
      POKE S(Q),170: FOR M=1 TO 200: NEXT:
      POKE S(Q),171: NEXT
1006  POKE S(121),170: FOR M=1 TO 200: NEXT:
      POKE S(121),171
1010  J=122: POKE S(J),170: G=0
1020  N=J+Z(G)/2:
      IF PEEK(S(N))=HL THEN POKE S(N),170:
      POKE S(J),171: J=N: G=G-2+4*(G<2):
      GOTO 1070
1021  IF PEEK(S(N))=171 THEN POKE S(N),[illegible]
      POKE S(J),16[illegible]: J=N: G=G[illegible]
      GOTO 1[illegible]
1025  IF PEEK(S(N))=1[illegible] THEN POKE [illegible]
      POKE S(J),171: PRINT CHR$[illegible]
      FOR M=1 TO 1[illegible]: NEXT: [illegible]
      GOTO [illegible]
[illegible] G=G-1+4*(G[illegible]): GOTO 10[illegible]
2[illegible] N=J+[illegible]2:
      IF PEEK(S(N))=171 THEN POKE S(N),170:
      POKE S(J),4[illegible] GOTO [illegible]
2045  IF [illegible] THEN PRINT [illegible]
      FOR Q=[illegible] TO [illegible]
      POKE [illegible]
      NEXT: [illegible]
2050  IF PEEK(-16384)=1[illegible] THEN [illegible]
2060  IF PEEK(-16384)=2[illegible] THEN HOME: END
2070  G=(G+1)*(G[illegible])
2080  POKE-16368,0: GOTO 20[illegible]
```

ZX81

Gent. redazione,
sono uno studente di ventun anni, iscritto al terzo anno di Matematica e possessore da qualche mese di un Sinclair ZX81.

Rispondendo al vostro cortese invito, vi invio una versione per ZX81 e ZX80 nuova ROM del generatore di labirinti apparso sul terzo numero della vostra notevole rivista.

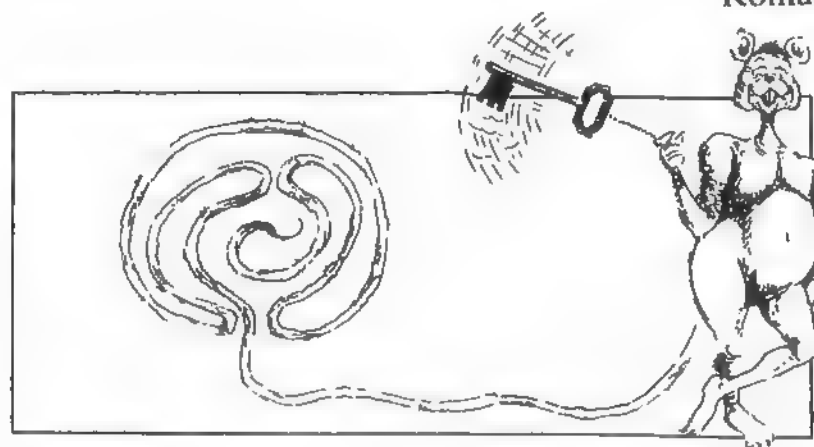
Le modifiche sono non molte, e in definitiva banali: sono stati adattati l'indirizzo d'inizio del display file (linea 110), i valori della tabella delle direzioni (linee da 20 a 50) e il codice dei caratteri di video inverso (linea 160) e blank (linea 210, 250). Inoltre, non consentendo lo ZX l'indice 0 nei vettori, si è dovuto cambiare il campo di variazione per J: ora J va da 1 a 4; di conseguenza si è modificato il suo incremento modulo 4 (linea 170).

Una nota: il labirinto generato dal programma così com'è è quello di dimensione massima, ottenibile senza rischi; chi volesse, può provare a far partire le righe dalla 0: deve però verificare che non vi siano dei 128 nella parte finale del programma. Per avere labirinti più piccoli, basta ridurre i valori limite per I

```
 10 DIM A(4)
 20 LET A(1)=2
 30 LET A(2)=66
 40 LET A(3)=-2
 50 LET A(4)=-66
 60 FOR I=2 TO 20
 70 FOR J=0 TO 30
 80 PRINT AT I, J; "■"
 90 NEXT J
100 NEXT I
110 LET A=PEEK 16396+256*PEEK 16397+101
120 POKE A,5
130 LET J=INT (RND*4+1)
140 LET I=J
150 LET B=A+A(J)
160 IF PEEK B=128 THEN GOTO 200
170 LET J=J*(J<4)+1
180 IF J <> I THEN GOTO 150
190 GOTO 240
200 POKE B,J
210 POKE (A+B)/2,0
220 LET A=B
230 GOTO 130
240 LET J=PEEK A
250 POKE A,0
260 IF J=5 THEN GOTO 290
270 LET A=A-A(J)
280 GOTO 130
```

e J nelle linee 60 e 70, senza altre modifiche. Se poi si vuole piazzare il labirinto in un altro punto dello schermo, o far partire la "costruzione" da un altro punto, avendo cura che sia interno e con coordinate entrambi dispari, occorrerà sostituire il valore 101 nella linea 110 con un $x=1+33*h+k$, dove h e k sono rispettivamente la riga e la colonna del punto di origine del labirinto.

Con i miei complimenti e cari saluti.

Federico Frezza
Roma

PICCOLI ANNUNCI

Commodore VIC 20

Scambio software per VIC 20. Cerco cartucce o altre espansioni (ROM, RAM). Inviare lista, invierò la mia. Leonardo Fei, via A. Fava 6, 20125 Milano, tel. 02-6894142.

Cambio gioco ROM Alien con altro gioco ROM per computer VIC 20. Attilio Muratori, via Roma 76, 47100 Forlì, tel 64221.

Vendo/acquisto/scambio programmi VIC 20 tramite invio cassetta ed eventualmente listato. Chiedere elenco o inviare cassetta. Risposta sollecita a tutti. Michele Capo, viale Petrarca 90, 57100 Livorno, tel. 402130.

Vendo giochi per VIC 20 a basso prezzo. Per ricevere la lista dei giochi spedire L. 350 in francobolli a Marco Vercellesi, via Mario Pichi 11, 20143 Milano, tel. 02-8394831.

Cambio per VIC 20 programmi di vario genere, e esperienze sulla grafica ad alta risoluzione. Maurizio Mellone, via Sabbionara 9, 36061 Bassano del Grappa (VI), tel. 0424-20015.

Vendo per VIC 20 breve programma per aumentare la capacità di schermo da 506 a 832 caratteri (+65%). Inviare L. 5000 in contanti a. Massimo Schianchi, via G. Miranda 3, 80131 Napoli

Vendonsi programmi gestionali per VIC: Condominio, Archivio clienti, word processing, programmi didattici; disponibili su disco o cassetta. Vendonsi fotocopie letteratura tecnica italiana e estera sul VIC 20. Francesco Del Vecchio, via Amoruso 34, 70124 Bari, tel. 510322.

Scambio programmi per VIC 20 e cerco possessori del medesimo per formare una banca programmi, ne possiedo circa 250; chi è interessato scriva a Lionello Zanella, via Virgilio 21, 74025 Marina di Ginosa (TA), tel. 099-627090.

Scambio/vendo programmi e informazioni con utenti VIC 20 anche su cassetta. Assicuro e chiedo massima onestà sia per lettera che per pacco. Attendo risposte anche per telefono. Gabriele Braga, via Bellaria 28, 40139 Bologna, tel. 545469.

Cerco per VIC 20 programma di uso per radio libera per gestione archivio dischi e cassette. Per ulteriori informazioni sulla quantità e sul tipo di gestione aspetto telefonata (ore pasti). Roberto Oselladore, via Corridoni 34/1, 30170 Mestre (VE), tel. 041-59896

Cambio programmi e idee per VIC 20. Massima serietà. Uso drive, superex, monitor assembler. Giovanni Ferrero, via Virgilio 4, 43100 Parma, tel. 494458

Cambio/vendo/compro software VIC 20 su cassetta Dispongo di ottimi giochi, prog utili, suono. etc. Per le liste inviate L. 1000. Fate offerte di vendita o scambio Rispondo a tutti Giorgio Ferrario, via Adua 1, 21052 Busto Arsizio (VA)

Compro programmi registrati su cassetta per VIC 20 Renato Bovenga, viale Miramare 75, 34100 Trieste, tel. 413205

Cambio programmi per VIC 20 su cassetta con altri diversi. Alberto Locatelli, via Sardegna 31, 20146 Milano, tel. 496576

Cambio/vendo software VIC 20 su cassetta. Dispongo di ottimi giochi, programmi utility, matematici, grafici, dimostrativi (anche con S. Expander). Biblioteca 100 programmi. Per lista inviare L. 1000. Carlo Comensoli, via S. Zenone 6/a, 25040 Demo (BS), tel. 0364-61389

Vendo word processor per VIC 20. L. 10000 su cassetta, L. 7000 listato. Funziona con stampante 80 colonne. Possibilità registrazione testo su cassetta. 10 comandi Piero Mellano, via Belvedere 78, 10028 Trofarello (TO)

Vendo/cambio programmi VIC 20 giochi, educativi. 3,5 K. Richiedere lista a Pautasso Francesco, via Cesare Battisti 22, 22070 Casnate (CO), tel. 031-451328

Cambio/vendo software per il VIC 20 a modestissimi pressi. Telefonare (sera) a Roberto Silva, via L. Cagnola 3, 20124 Milano, tel. 02-317228

Cerco software e riviste (anche in lingua straniera) per VIC 20. Vendo interfaccia per registratore VIC 1001. Contatto possessori VIC per scambio software vario. Davide Zegna, via Marco Polo 1, 17025 Loano (SV), tel. 019-670582

Cambio o vendo software per VIC 20 moltissimi programmi per gestione, tecnica, gioco; legge 373, equo canone, prospettiva, etc. Maurizio Mellone, via Sabbionara 9, 36061 Bassano del Grappa (VI), tel. 0424-20015

Vendo nastro con 5 giochi per VIC 20 a L. 7000 Massima serietà. Per informazioni telefonare 0971-582283 chiedere di Michele. Oppure scrivere a Michele Piscopo, piazza Marconi 9, 66013 Chieti.

Commodore PET/CBM

Cambio programmi per PET 4032 su cassette. Inviare lista, invierò la mia. Gianni Musci, via Lodi 65, 20139 Milano, tel. 02-5690238

Vendo/scambio per PET/CBM Commodore prog vari: RTTY, Mailbox, QRB Locat., gestionali, Ingegneria per Apple RTTY, Mailbox, CW, Compilatore tasch (cerco prog. paghe, stipendi, compilatore per CBM/PET). Paolo (IW6MEQ) Stella, via N. Moscardelli 28, 67100 L'Aquila, tel. 23273

Vendo ROM per CBM 40/8032-3032 Basic 4. Aggiunge 16 com. al Basic tra i quali: contr. cursore, sort alfanumerico, somma, sottraz., moltipl in tripla precisione (24 cifre), puntegg. autom. numeri, input controll., etc. Specificare se ROM 9000 o A000. Costo con istruz. L. 200.000. Giuseppe Mannino, viale Primavera 3/6, 16148 Genova, tel. 010-332827

Vendo programma completo per relazione geotecnica di calcolo dei cedimenti di fondazioni superficiali poggianti su terreni stratificati per CBM 3032 Commodore versione video e stampante Ing. Alvaro Albani, via Castelfidardo 7, 47037 Rimini (FO), tel 0541-25765

Vendo o cambio programmi di ogni genere per PET/CBM. Richiedete l'elenco Andrea Gambedotti, via Campo sportivo 12, 12032 Barge (CU) tel. 0175-926248

Vendo programmi paghe, contabilità generale e semplificata, 373, condomini, alberghi, ottici, studi medici, fatturazione e magazzino, word processing per PET/CBM 3032/4032. Tel. 0972-31669 dalle 14 alle 19. Alfredo Casciano, via Mons. Virgilio 105, 85329 Venosa (PZ)

Cambio programmi di tutti i generi per PET/CBM serie 3000. Se possibile inviare lista. Non rispondere se non intenzionati. Prometto pronta risposta a tutti. Andrea Chiuppi, via Diocleziano 41, 33010 Feletto Uberto (UD), tel. 0432-681479

Vendo giochi per PET/CBM Basic 4.0 vasto assortimento Richiedere listino con invio di L. 800 in francobolli. Scambio anche giochi ma solo con altri di uguale valore e interesse. Antonio Di Gilio, via Monte Cervino 1, 30030 Favaro Veneto (VE), tel. 041-611259

Cambio software per PET serie 4000-8000 su disco o cassetta. Dispongo di oltre 500 programmi tra gestionali e giochi. Rispondo a tutti. Cerco data base a campi variabili su floppy 4040-8050. Giovanni Nuvoli, via Ulumos 3, 07018 Pozzomaggiore (SS), tel. 079-801276

Vendo cassetta programmi sonori PET/CBM· Pronostici totocalcio, Autopista, Blackjack, Battaglia navale, Tombola, Pallina. L. 25 000+SP. Posseggo programmi pubblicati su riviste sonorizzati e corretti. Luigi Cuomo, via Filangieri 72, 86095 Frosolone

Vendo software civile PET stampa, anche su cassette, supertestato zona sismica, telaio, fondazioni, grigliati, verifiche sezioni, 373, ecc. Ing Giovanni Gaviani, via Finelli 3, 40100 Bologna, tel. 051-230126

Sinclair ZX80-ZX81 Spectrum

Vendo oltre 30 programmi (Corse dei cani, Master mind e altri 28) per ZX80 nuova ROM e ZX81 per sole L. 9000 già registrati su cassetta Livio Pomi, via B. Giacomini 2, 21051 Arcisate (VA), tel. 470343

Compro riviste sui computer inglesi ed americane ed inoltre software, listati di qualsiasi genere e cassete per ZX81. Vita Luciano, via Oreste Pennati 1, 20052 Monza (MI), tel. 039-367029

Accanito gruppo di programmatori ha organizzato un'iniziativa che in Italia non ha uguali mettiamo a disposizione di tutti coloro che ci scriveranno una vastissima biblioteca di programmi inediti che girano sullo ZX80-81. Per accedervi basta spedire almeno un programma con L. 1000 in francobolli specificando la marca del Vs. computer e le Vs. preferenze; vi rispediremo minimo 3 programmi Scrivete a questo indirizzo· Reggiani Franco, via Zabarella 15, 32028 Piove di Sacco (PD)

Cerco sinclairisti nella sola zona di Napoli per scopo fondazione club. Emilio Triunfo, via Cumana 9, 80126 Napoli, tel 081-633274

I lettori che voglio vendere, comperare o scambiare software, o desiderano dare informazioni possono compilare il tagliando pubblicato in fondo a questa rubrica. Il servizio è gratuito. La redazione si riserva il diritto insindacabile di rifiutare, sospendere o modificare qualsiasi inserzione.

Gli annunci sono riservati ai privati o ai club senza scopo di lucro.

Daremo la precedenza agli annunci che si riferiscono a software, programmi, libri e riviste, club per personal computer.

Un annuncio sarà più efficace se seguirete queste indicazioni:

- La *prima parola* deve essere esplicativa del vostro messaggio: scambio, vendo, compro, cerco... Per renderla più evidente la stamperemo in corsivo.
- Nel testo, riferitevi ad *un solo tipo* di computer (VIC 20, ZX80,...). L'annuncio apparirà sotto la testata relativa a quel computer. Se volete fare un annuncio per due o più computer, compilate due tagliandi.
- Date il vostro *recapito* con esattezza: nome e cognome, via e numero, *cap* e località, provincia, *prefisso* e numero telefonico.

Vendo packages software di ingegneria civile e calcolo strutture in zona sismica per Sinclair ZX81 16 K. Carlo Conticelli, largo G.I. Molina 4, 40138 Bologna, tel. 051-305458

Idee nuove per lo ZX81. Questo mese: ZX HiFi test. Per ZX81 senza modifiche o aggiunte hardware anche solo 1 K RAM!! Per informazioni e prenotazioni: Dionisio Castello, via Basilicata 15, 04019 Terracina (LT)

Software ZX81 vendo programma per fatturazione, preventivi, conti e listino prezzi e articoli a L. 30000 in cassetta. Tel. (059) 683923 ore pasti. Ivano Pongiluppi, via Roosevelt 63, 41012 Carpi (MO)

Sinclair club costituito da utenti ZX per scambio idee, programmi e esperienze hardware. L'adesione dà diritto alla ricezione di un bollettino trimestrale e a facilitazioni varie. Quota annuale L. 18000. Sinclair Club, via Molino vecchio 10/F, 40026 Imola (BO)

Vendo cassetta scacchi alta risoluzione, 10 livelli, alta velocità per ZX Spectrum 48 K. Per chi avesse il 16 K, lo converto in 48 K per 120000 lire. Dante Vialetto, via Gorizia 5, 21053 Castellanza (VA), tel. 0331-500713

Vendo cassetta linguaggio Pascal completa di manuale per Spectrum 48 K a L. 40000. Per chi avesse solo il 16 K lo posso convertire in 48 K per L. 120000. Dante Vialetto, via Gorizia 5, 21053 Castellanza (VA), tel. 0331-500713

Vendo per ZX81 programma Tartinville (per ricerca capisaldi in equazione parametrica di 2° grado) a L. 15000 tutto compreso (cassetta+spese spedizione). Posseggo inoltre programmi di vario genere. Bruno Cardella, via Calabria 4, lotto 43, 90100 Palermo, tel. 512302

Programma Pert per ZX80/81 ottimizzazione progetti sino a 350 attività, ottima documentazione in italiano con teoria ed esempi, offro a L. 25000 compresa spedizione. Scrivete per informazioni a Giovanni Servi, via Giovanni XXIII, 41012 Carpi (MO)

Cerco programmi per ZX81 e in special modo gli scacchi. Scambierei con i miei programmi. Vendo inoltre cassette 2 giochi per Sinclair 16 Kbyte molto validi e a prezzi modesti. Telefonare o scrivere a Andrea Lombardo, corso Sempione 39, 20145 Milano, tel. 02-382897

Sinclair club costituito per scambi di software, idee, schemi hardware per ZX. Preparazione di un bollettino trimestrale per i soci. Arrigo Bondi, via Molino Vecchio 10/F, 40026 Imola (BO)

Programmi rifinitissimi originali fantastiche novità assolute ZX81/80 8 K. Due cassette TDK 90 minuti zeppe ambo i lati L. 18000 ciascuna. Tutte due, totalmente diverse L. 34000. Incredibile: ogni programma vi costa meno di 500 lire! Garantisco assoluta soddisfazione. Bruno Del Medico, via Torino 72, 04016 Sabaudia (LT)

Vendo soft a 1 K per ZX81. Eseguo montaggi di kit espansione per ZX80-81 a prezzi interessanti. Scambio informazioni su computer Sinclair. Francesco Buemi, via G. Barbereschi 201/5B, 16149 Genova, tel. 010-267120

Vendo programma per calcolo QRB per ZX81 registrato su cassetta adatto sia per stampante che video a L. 15000. Inoltre vendo programma per controllo contest dei QSO doppi L. 15000 su cassetta. Leandro Iaccarino, via Vanassina 2/A, 80073 Capri, 081-8379146

Vendo per ZX81 programmi originali inglesi: Packman, Defender 3D, Simulazione volo, Defender, Zombies, vasto assortimento. Oltre 100 voci. Inviare L. 2000 per listino completo. Stefano Nocilli, via Giuseppe De Leva, 00179 Roma

Scambio programmi per computer ZX81, ZX80 nuova ROM, ZX Spectrum, cassette, software, informazioni e dati tecnici. Scrivere o telefonare ore pasti. Armando Pavese, via Cottolengo 59, 13051 Biella, tel. 015-27353

Per ZX81 programmi assolute novità: Scacchi II, Computacalc, Calculex II, Mazogs, Scramble, Mazeman, Asteroids, Galaxian a L. 9000/15000 cad. Elenco gratis a richiesta. Vendo anche espansione 32 K assemblata in contenitore metallico e garantita L. 150000. Massimo Soncini, via Monte Suello 3, 20133 Milano, tel. 02-727665

Vendo o scambio programmi per ZX81 da L. 3000 a 5000 (Nim, Impiccato, Slot machine, Explorer, Decidere, Combattimento su Londra e tanti altri. Linguaggio macchina e Basic). Scrivere o telefonare per lista. Marco Carrubba, via M. Campionesi 29, 20135 Milano, tel. 02-585294

Vendo programmi per ZX81-ZX80, veramente nuovi, inediti. Telefonare o scrivere a Stefano Ricci, via Fanny Tacchinardi 21, 00168 Roma, tel. 06-6274831

Compro ROM assembler e Basic per ZX 80 o ZX81 Sinclair. Cerco principalmente il listato di queste due ROM. Scrivere per accordo a: Giosuè Caliano, via L. Guercio 150, 84100 Salerno, tel. 089-399750

DAI

Vendo copie del "DAI Firmware Manual" contenente elenco POKE e PEEK possibili e mappe complete di memoria e listato sorgente del Basic implementato sul DAI P.C. 48 K, commentato, 250 pag., rilegato. Roberto Porta, corso Cavallotti, 27, 15100 Alessandria, tel. 0131-63016

Cerco utenti di computers DAI disposti a scambiare idee e programmi circa il suddetto apparecchio. Per accordi scrivere o telefonare a: Stefano Filippi, via Monte Grappa 5, 32030 Fonzaso (BL), tel. 5165

Vendo per DAI 48 K fantastici programmi garantiti inediti con colore e suono. 10 programmi su cassetta (Invasori, Midway, Reverse, etc.). L. 20000 contrass. 1700 in più. Dispongo anche centinaia di programmi per Sinclair a prezzo bassissimo. Per informazioni allegare francobollo. Avena Vincenza, via Garibaldi, 04016 Sabaudia (LT)

Esperto programmatore C.N.R. e allievo accademia d'arte vendo per DAI P.C. (e per altri P.C. per chi possa contattarmi) cassette con mie opere di computer art (astratto e figurativo). Scrivere per informazioni e prezzi allegando L. 1000 a: Alberto Pollastri, via di Pratale 28B, 56100 Pisa, tel. 050-20584

Vendo per DAI software e documentazione. Scacchi, Assembler, Text editor, Caratteri mode 5, giochi, grafica, simulazione, DCE bus, firmware ROM, schemi elettrici. Salvatore Pennisi, via Mario Borsa 63, 00159 Roma, tel 06-4387248

Apple II

Vendo programma per calcolo muro generale in cemento armato per zone sismiche e non sismiche per Apple II. Crescenzio Gallo, via Indipendenza 51, 71041 Carapelle (FG), tel. 0885-95242

Scambio programmi di qualsiasi tipo per Apple II (16 K). Inviate la vostra lista ed io vi risponderò con la mia. Antonio Curci, via Nuovo Ponte 40, 84086 Roccapiemonte (SA)

Cambio/vendo i più famosi package professionali CP/M e non per Apple II: potenti data base relazionali, linguaggi, word processing. Luca De Matteis, via S. Lavagnini 26, 50129 Firenze, tel. 055-474739

Hei! Mio fratello ha un Apple II che usa per lavoro. Io ci gioco e mi piacerebbe cambiare i miei programmi con i vostri. Ho 14 anni. Francesca Melani, v.le Ippolito Nievo 100, 57100 Livorno

Cambio/vendo software per Apple II, scientifici, gestionali, utilità, giochi, vasta scelta. Inviare la lista a cui risponderò con la mia. Massimo Bracci, via della Repubblica 10, 56030 Montecalvoli (PI), tel. 0587-748042, ore pasti.

Vendo programma strutturato analisi edifici in cemento armato (telai e/o mensole) sollecitati da carichi permanenti e/o distorsioni termiche e forze sismiche statiche o dinamiche. Apple II. Giovanni Viola, via Pietragrossa1, 66100 Chieti, tel. 0871-67477

Compro, cambio, vendo software per Apple II. Claudio Citarella, via Parroco Federico 41, 80045 Pompei (NA), tel 081-8632946

Cambio programmi e giochi per Apple II. Cerco nuovi sistemi word processing per Apple II. Salvatore Gentile, via Quarto 11/1, 16147 Genova, tel. 010-388687

Scambio programmi per Apple II. Scrivere a Giuseppe Milisich, viale E. Caldara 13/3, 20122 Milano

HP 85/87

Cerco possessori di HP87/85 per scambio idee e programmi. Giorgio Scaglianti, via Ferrara 10, 44034 Copparo (FE)

Cerco lettore esperto disposto tradurre in linguaggio HP-85 programmi pubblicati su questa rivista. Per accordi scrivere a Francesco Piccione, via delle Querce 32, 95030 Gavina di Catania

Scambio/vendo programmi ingegneria civile edile predisposti per HP85 con ROM matrix e ROM programmazione avanzata e stampante esterna Centronics 150 cps bidirezionale. L'HP ha espansione a 32 K e supporto esterno su nastri. Sergio Andruzzi, via Paolo Bentivoglio 13, 00165 Roma, tel. 06-6376113

Atari

Vendo e cambio per Atari 400-800 molti programmi gioco e utility originali americani su cassetta o disco. Marcello Guidotti, via Cutilia 27, 00183 Roma, tel 06-778896

Cambio/vendo Atari software escluso programmini copiati da riviste. Antonio Sciarra, via Lambro 1D, 00199 Roma, tel. 06-8451572/867869

Svendo per Atari 400-800 a prezzo di realizzo cartucce giochi e linguaggi complete di manuali di istruzioni. Telefonare a Giorgio al 055-252278. Giorgio Lapi, via Scandicci Alto 30, 50018 Scandicci (FI)

Compro, pagando bene, programmi per Atari 400 sia utility che di game. Andrea Verona, via Mascheroni 12, 20145 Milano, tel. 02-495814

Texas TI 99/4A

Cerco utilizzatori per TI 99/4A in Pavia per creazione club novantanoviano, scambio software e idee di hardware con amici del TI 99 di tutta Italia. Uberto Zaga, via Ferrini 77, 27100 Pavia, tel. 470367

Vendo oppure cambio cartuccia per home TI 99/4A video games 1. Contiene 3 giochi in inglese, francese e tedesco: flipper, tiro a segno, domino con velocità, livello etc. a L. 50 000, compro anche programmi. Adriano Sarzina, via Bellini 17, 25077 Roè Volciano (BS)

Scambio diagrammi di flusso di tutti i generi e tipi, vendo e compero programmi per calcolatrici serie TI della Texas (programmabili 53—59), miglioro i programmi per le stesse e cerco amici per scambio informazioni. Andrea Rimicci, via delle Fornaci 12/1, 17011 Albisola Capo (SV)

Vendo traduzioni dall'inglese dei manuali dei moduli di comando SSS "Video Chess" (L. 3.000) e "TI Invaders" per computer TI 99/4A. Cerco inoltre possessori di 99/4A per scambio di idee e software. Filippo Cerulo, via Mercato 9, 82038 Vitulano (BN)

Cerco programmi di tutti i tipi ma specialmente giochi per il personal computer Texas TI 99/4A. Acquisto programmi scritti o in cassetta. Telefonate offerte ore pasti. Renato De Momi, via G. Bertacchi 3/A, 35100 Padova, tel 049-758328

HO il computer della Texas Instruments TI 99/A e sono alle prime armi in fatto di programmazione. Cerco programmi possibilmente giochi per detto computer. Possibilmente scritti o in cassette. Inviare lista e prezzi. Adriano Modolo, viale S. Marco 134, 30173 Mestre (VE), tel 955831

Cerco programmi e giochi per home computer TI 99/4A da acquistare. Sandro Magni, via Vetta d'Italia 2, 20052 Monza, tel. 039-735208

Cambio software per TI 99/4A. Mi interessa tutto. Speditemi la lista dei programmi, vi manderò i miei. Lorenzo Tomellini, via C. Ferrari, 20015 Parabiago (MI)

Acorn Atom

Cerco utenti Atom Acorn per scambio idee e programmi. Carlo Giuliano, via S. Agostino 193, 56100 Pisa, tel 40261

Scambio esperienze e programmi per l'Atom Acorn che è in mio possesso. Preferirei programmi di giochi, utilities, animazione grafica e vari. Amici dell'Atom, perché non creiamo un club? Giosuè Caliano, via L. Guercio 150, 84100 Salerno, tel. 089-399750

Sharp

Vendo software di vario genere per Sharp M280B. Programmi di tutti i generi. Disposto anche ad effettuare eventuali scambi. Prezzi veramente mai visti! Affrettatevi! Stefano Lazzaro, via Monte Sabotino 2, 35100 Padova, tel 049-22675

Vendo per Sharp MZ-80 K la serie completa dei programmi per lavorare in linguaggio macchina, composta da Relocatable Loader+Assembler+Symbolic Debugger+Text editor+Machine language, su cassetta, corredati dei tre relativi manuali per l'uso. Il tutto L. 120 000. Giovanelli Claudio, via Ripamonti 194, 20141 Milano, tel 02-536926

TRS-80

Cerco hobbisti disposti a scambiare esperienze su TRS 80 o VGS. Mi interessano applicazioni radioamatoriali. Cerco linguaggi residenti su cassetta. PO BOX 154, 05100 Terni

Osborne I

Cercasi utilizzatori Osborne I per scambio software e notizie esperienze. Paolo Medici, via S. Cafasso 21, 10132 Torino, tel. 011-898739

Triumph Adler

Sono in possesso di un Triumph Adler mod. P2. Cerco programmi per l'elaborazione mod. 740. Cerco inoltre altri programmi, giochi, da inserire in questo computer. Nerio Culpo, via F. Baracca 9, 22060 Carugo (CO), tel. 762308

Cerco utenti di Alphatronic P2 per scambio di informazioni, programmi etc. Gaetano Celentano, via Orza 3, 84087 Sarno (SA), tel. 081-941285

CP/M

Software sotto CP/M e per Olivetti M20, già pronto e/o su commissione. Prezzi molto convenienti. Richiedete il catalogo gratis a: Mario Magnaghi, via Beldì 19, 28068 Romentino (NO)

Metto a disposizione un computer CP/M con 48 K RAM e interfaccia 8+8 bit I/O ed 1 scheda MC6800 con I/O parallelo per effettuare esperienze e studi di robotica con altre persone interessate. Graziano Ceccotti, via Livornese Est 124, 56030 Perignano (PI), tel. 0587-616046

Cobol

Vendo sorgenti di programmi gestionali in linguaggio Cobol, già ammortizzati e collaudati con relativi manuali di: contabilità generale e adempimenti IVA a L. 800000, fatturazione immediata e differita con provvigioni agenti e ricevute a L. 400000, gestione magazzino commerciale a L. 500000, e altri programmi gestionali. Telefonare ore 19-20 allo 0376-534087. Dino Cominotti, via Saliceto 18, 46029 Suzzara (MN)

Riviste

Vendo arretrati "Elettronica oggi", "Elettronica pratica", "Nuova elettronica", "Radio kit Elettronica" agli attuali prezzi di copertina. Compro "Bit" n. 3 ed "Informatica oggi" n. 4 e 7/8 Marco Fabbri, via C. Rocco 48, 80043 Madonna de l'Arco (NA), tel. 081/8981107

Diversi

Vendo metodo anti-list per Apple, PET, PET old ROM, VIC 20. Inoltre vendo o cambio software per suddetti e TRS80, ZX80/81, Atari, HP, TI58C/59, PC1201. Mando lista a chi telefona o scrive allegando L. 500. Paolo Marcato, via C. Battisti 3, 35027 Noventa Padovana (PD), tel. 049-502475

Compro listati di programmi per il PET 3032 e ZX80/81 oppure altri che si possono adattare al PET. Mandate informazioni e mettetevi in contatto con Antonio Muslo, viale di Levante 64, 71042 Cerignola (FG), 0885-29316

Cambio/vendo software per Apple II e TRS-80 mod. III. Giovanni Carrella, via Felline 11/C, 84100 Salerno, tel. 089-354854

Vendo splendidi programmi per VIC 20, Tandy, Apple II, ZX80/81, PET/CBM, TI 58/59, DAI. Inviare L. 500 per la lista del software che vi interessa a: Giorgio Ferrario, via Adua 1, 21052 Busto Arsizio (MI)

Vendo software di ogni tipo per personal VIC 20 e Sharp. Posso inoltre procurare fotocopie di manuali del VIC 20. Scrivere o telefonare a Roberto Minoia, via Ippocastani 10, 20152 Milano, tel. 02-4595642

Vendo software civile PET ZX81 completissimo, pluritestato, manuali, verifica sez. telaio piano, fondazioni, muri, 373, etc. (solo PET) zona sismica, analisi dinamica, etc. Ing. Giovanni Gaviani, via Finelli 3, Bologna, tel. 051/230126

Hardware

Vendo stampante per ZX81+4 rotoli carta+espansione 16 K originale a sole L. 250000. Gianni Fassi, via P.pe Eugenio 65, 00185 Roma, tel. 7316539

Cerco ZX81 o ZX80 usato ma perfettamente funzionante senza altri accessori o programmi e a prezzo veramente conveniente. Ernesto Boracchi, via Ronchetti 2, 21013 Gallarate (VA), tel. 0331-795649

Cambio 3 alimentatori professionali Leybold nuovissimi (0-300 V con più uscite indipendenti) + analizzatore di corrente Leybold (valore totale dei 4 apparecchi L. 1300000) con stampante o plotter. Rocco Macrì, via S. Pietro Campagna 154, 06081 Assisi (PG), tel. 812379

Cambio ZX80+16 K con monitor. Compro, vendo, cambio software TI99/4A. Salvatore Sbacchis, via Don G. Minzoni 2/E, 90143 Palermo, tel. 091-547670

Vendo VIC 20, espansione alta risoluzione×3 K RAM, espansione linguaggio macchina, cartridge "Alien", joystick, paddle, libri, moltissimi programmi, il tutto a prezzo strepitoso! Telefonare ore pasti. Alessio Camplone, via Martiri della Resistenza 42, 60125 Ancona, tel. 071-899520

Vendo Centronics 737 con manuale originale e software di utility. Enrico Stomeo, C.P. 654, 30100 Venezia, tel. 041-762042

Vendo CBM 3022 floppy disk da 1 megabyte computing stampante 3032 interfaccia parallela RS232 modem acustico e 200 programmi con Basic Plus e WPP al miglior offerente. Pier Giorgio Perron Cabus corso S. Martino 4, 10122 Torino, tel. 530656

Vendo VIC 20+unità cassette Commodore a sole L. 500.000. Il tutto è nuovo e ancora imballato. Claudio Giacometti, via A. De Gasperi 22, 45026 Lendinara (RO), tel. 0425-61871 (ore ufficio)

Vendo HP-41 C completo pile e alimentatore corrente, lettore schede, stampante, 3 moduli memorie e relativo contenitore, custodia, manuali istruzioni per 41 C, manuale stampante e lettore, guida programmazione, tutto in italiano. L. 1200000. Possibilità prezzo speciale per schede vergini, contenitori programmi e libri HP Rotoli carta termica e copie rivista club HP. Telefonare ore ufficio. G. Willy Aletti, via Tiziano 9/5, 20145 Milano, tel. 02-6073971

Vendo espansioni 16 - 4 - 32 K per ZX80 e 81, generatore di caratteri a L. 38000 kit slow kit per ZX80 a L. 25000 modulo porte input/output cassette giochi a L. 12000 tutto per ZX. Pagamento anticipato. Alberto Bassi, via M. Bonavita 35, 47100 Forlì, tel. 0543-24345

Vendo anche separatamente a miglior offerente TRS-80 mod. I lev. 2 con 16 K RAM, video verde e tastierino numerico+expansion interface con 32 K RAM+stampante Centronics 770+software vario. Telefonare lun -ven. ore pasti a Franco. Francesco Fedele, via Antagora 26, 00124 Roma (Casalpalocco), tel. 6094745

Vendo ZX80+8 K ROM+16 K RAM+alimentatore+cavi+programmi tutto in ottime condizioni a L. 350000 tratt Telefonare ore pasti Stefano Lazzaro, via Monte Sabotino 2, 35100 Padova, tel. 22675

Vendo CBM 4040 floppy disk usato pochissimo. Vendo anche progr per gestione archivio dati (log. radioamatori, elenchi clienti, telef. biblioteca ecc.) con defin. voci, ricerca veloce dati, aggiorn. ecc. (per 8032). Roberto Vendrame IN3VRR, via Maso della Pieve 72, 39100 Bolzano, tel. 0471-940615 (ore pasti), 41333 (int. 286 ufficio)

PICCOLI ANNUNCI PERSONAL SOFTWARE

Sei un lettore di PERSONAL-SOFTWARE e vuoi entrare in contatto con tutti gli altri lettori per comperare, cambiare o vendere software? Spedisci questo tagliando a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome Cognome

Via CAP

Città Tel

Servizio programmi

Per alcuni dei programmi pubblicati, *Personal Software* mette a disposizione dischi e nastri già registrati, realizzati in collaborazione con l'autore. Potete ottenerli in contrassegno, pagando direttamente al postino la cifra indicata, spedendo il tagliando pubblicato in fondo alla pagina.

N.	Sistema	Programmi	Supporto	pubblicato in *Personal Software* n.	Prezzo
1	Apple II+	La carta del cielo Collisione	floppy 5" DOS 3.3	3 pag. 83 3 pag. 93	30.000
2	TRS-80 mod. I	Backgammon	floppy 5" DOS 2.3	3 pag. 89	25.000
3	PET/CBM 3032/4032	Editor/Assembler in Basic	floppy 5" 3032/4032+3040/4040	2 pag. 33	40.000
4	Apple II+	Interi in precisione multipla	floppy 5" DOS 3.3	4 pag. 17	40.000
5	PET/CBM 3032/4032	Gioco del calcio	floppy 5" 3032/4032+3040/4040	4 pag. 67	25.000

Spedire in busta chiusa a

PERSONAL SOFTWARE
Servizio Programmi
Via Rosellini 12
20124 Milano

Inviatemi i seguenti nastri e/o dischi con i programmi pubblicati su *Personal Software*
n. ______________________

per un totale di lire ______________________
che pagherò al postino alla consegna del pacco.

____
Cognome e nome

____
Indirizzo

____
Cap., Località

____
Firma

UNA PUBBLICAZIONE
DEL GRUPPO EDITORIALE JACKSON

ANNO 2 N. 4 GENNAIO-FEBBRAIO 1983

DIRETTORE RESPONSABILE: Giampietro Zanga

DIRETTORE: Mauro Boscarol

REDAZIONE: Completo Software - Padova

HANNO COLLABORATO A QUESTO NUMERO:
E.M. Albani, M. Cerofolini, J. Commander, E. Ferreguti, A. Filz, M. Pelczarski, C. Saraceno, C. Sintini.
Copertina: Roberto Cortivo
Grafica: Carlo Buffa
Fotocomposizione: Composizioni Grafiche - Padova
Traduzioni: F. Santini, S. Ventura

CONTABILITÀ: Franco Mancini, Roberto Ostelli, Mariella Luciano, Franca Anelli, Sandra Cicuta, Gabriella Napoli

DIFFUSIONE E ABBONAMENTI: Luigi De Cao, Adela Bel Lozano, Ombretta Giannetto

AUTORIZZAZIONE ALLA PUBBLICAZIONE: Tribunale di Milano n. 69 del 20/2/1982

PUBBLICITÀ: Concessionario per l'Italia e l'Estero Reina s.r.l. Via Washington, 50 - 20146 Milano
Tel. (02) 4988066/7/8/9/060 (5 linee r.a.)
Telex 316213 REINA I

STAMPA: Arti Grafiche "La Cittadella" S.p.a.
Pieve del Cairo (PV)

Concessionario esclusivo per la DIFFUSIONE in Italia e all'Estero:
SODIP - Via Zuretti, 25 - 20125 Milano

Spedizione in abbonamento Postale Gruppo III/70
Prezzo della rivista L. 3.500. Numero arretrato L. 6.000.
Abbonamento annuo (10 numeri) L. 30.000; per l'Estero L. 48.000
I versamenti vanno indirizzati a: Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano - mediante emissione di assegno bancario, cartolina vaglia o utilizzando il c/c Postale numero 11666203.
Per i cambi di indirizzo, indicare, oltre naturalmente al nuovo, anche l'indirizzo precedente, ed allegare alla comunicazione l'importo di L. 500, anche in francobolli.

GRUPPO EDITORIALE JACKSON Srl

DIREZIONE, REDAZIONE, AMMINISTRAZIONE:
Via Rosellini, 12 - 20124 Milano - Telefoni: 68.03.68 - 68.00.54

SEDE LEGALE: Via Vincenzo Monti, 15 - 20123 Milano

DIREZIONE EDITORIALE: Giampietro Zanga e Paolo Reina
COORDINAMENTO EDITORIALE: Daniele Comboni

crescere.

Apple ha introdotto il concetto di personal in tutto il mondo. E in tutto il mondo Apple cresce. Cresce anche in Italia dove la Iret, che lo importa e ne cura l'assistenza, può oggi annunciare l'esistenza di una rete di vendita di oltre 300 centri specializzati che fanno di Apple il loro cavallo di battaglia.
E naturalmente crescono le vendite di Apple, perché il personal computing conquista piccole aziende, professionisti e privati. È facile prevedere quindi che Apple continuerà a crescere, anche perché l'unica cosa di Apple che non cresce sono i prezzi. (Chiedete l'offerta speciale ai nostri rivenditori).

apple® Il Personal Computer

Via Bovio, 5 - 42100 Reggio Emilia - Tel. 0522/32643 - TLX 530173 IRETRE

response